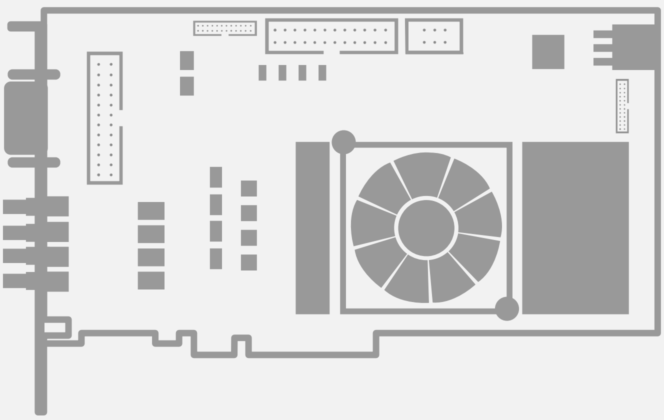
FUNCTIONAL GUIDE

Coaxlink

Coaxlink 12.6 Functional Guide

1629 Coaxlink Duo PCIe/104-EMB
1630 Coaxlink Mono
1631 Coaxlink Duo
1632 Coaxlink Quad
1633 Coaxlink Quad G3
1633-LH Coaxlink Quad G3 LH
1634 Coaxlink Duo PCIe/104-MIL
1635 Coaxlink Quad G3 DF
1637 Coaxlink Quad 3D-LLE
3602 Coaxlink Octo
3603 Coaxlink Quad CXP-12
3620 Coaxlink Quad CXP-12 JPEG
3621-LH Coaxlink Mono CXP-12 LH
3622 Coaxlink Duo CXP-12



Terms of Use

This documentation is provided with Coaxlink12.6.2 (doc build 2114).
www.euresys.com

Contents

1. About This Document

1.1. Document Scope

This document describes and explains how to use the functions of the Coaxlink products when they are operated with Coaxlink driver version 12.6.2

Unless specified, the functions described in this document are applicable to all the Coaxlink products and their firmware variants supported by the Coaxlink Driver.

1.2. Document Changes

2. Coaxlink Architecture

2.1. Main Elements

Quick overview of the main functional elements of a Coaxlink-based image acquisition system.

GenTL Hierarchy

Each functional element of Coaxlink is configured and controlled by GenICam features belonging to a GenTL module.

At the top of the hierarchy, there is one *GenTL System Module* per Host PC. It binds all the *GenTL Interface Modules* of a Host PC.

There is one *GenTL Interface Module* for each Coaxlink card. It binds all the *GenTL Device Modules* of a Coaxlink card.

There is one *GenTL Device Module* for each camera (or imaging device) attached to a Coaxlink card. The elements belonging to the imaging device (camera) itself are referred as *Remote Device*. By opposition, the elements belonging to the frame grabber are also referred as *Local Device*.

> **NOTE**
> The maximum number of cameras that can be attached to a Coaxlink card is determined by the installed firmware variant.

There is one *GenTL Data Stream Module* for each data stream delivered by a camera attached to a Coaxlink card. It gathers the elements involved into the image build-up and transport from the imaging device to a pool of GenTL buffers.

> **NOTE**
> The maximum number of data-stream for a camera attached to a Coaxlink card is determined by the installed firmware variant.

There is one *GenTL Buffer Module* for each image buffer.

Interface Module Main Elements (Orange)

I/O Lines

The "General Purpose I/O" on page 116 block gathers all the I/O ports of the card.

I/O Toolbox

The "I/O Toolbox" on page 133 block gathers a collection of tools used to build event streams from trigger and encoder devices attached to the I/O port inputs.

NOTE
These elements are common to- (or can be shared by-) all the GenTL Device Modules managed by the Coaxlink card

Device Module Main Element (Green)

Camera and Illumination Controller

This block is used to control the camera cycle and the illumination strobe. It can be configured to receive real-time (Camera) Cycle trigger events from any I/O Toolbox output stream. It produces two real-time signals: the Camera Trigger signal, sent to the camera trigger input, and the Strobe signal, sent to the illumination device associated with the camera.

NOTE
This element is common to- (or can be shared by-) all the GenTL Data Stream Modules related to that imaging device.

Data Stream Module Main Elements

Image Acquisition Controller

This block is used for the control of the acquisition gate. It can be configured to receive real-time start-of-scan and end-of-scan trigger events from any I/O Toolbox output stream.

Acquisition Gate

The "Acquisition Gate" on page 62 controls the data extraction and filters out the image data that doesn't need to be acquired.

FIFO Buffer

The on-board"FIFO Buffer" on page 61 temporarily stores the raw image data together with related metadata such as image size, pixel type, time-stamp...

Pixel Processing

The"Pixel Data Processing" on page 63 performs on-the-fly pixel processing.

Data Formatting

"Pixel Component Unpacking" on page 69, "Pixel Component Re-Ordering" on page 71

, "Endianness Conversion" on page 72 and "Pixel Ordering" on page 73 operations are configured according the desired pixel output format.

Image Data Transfer

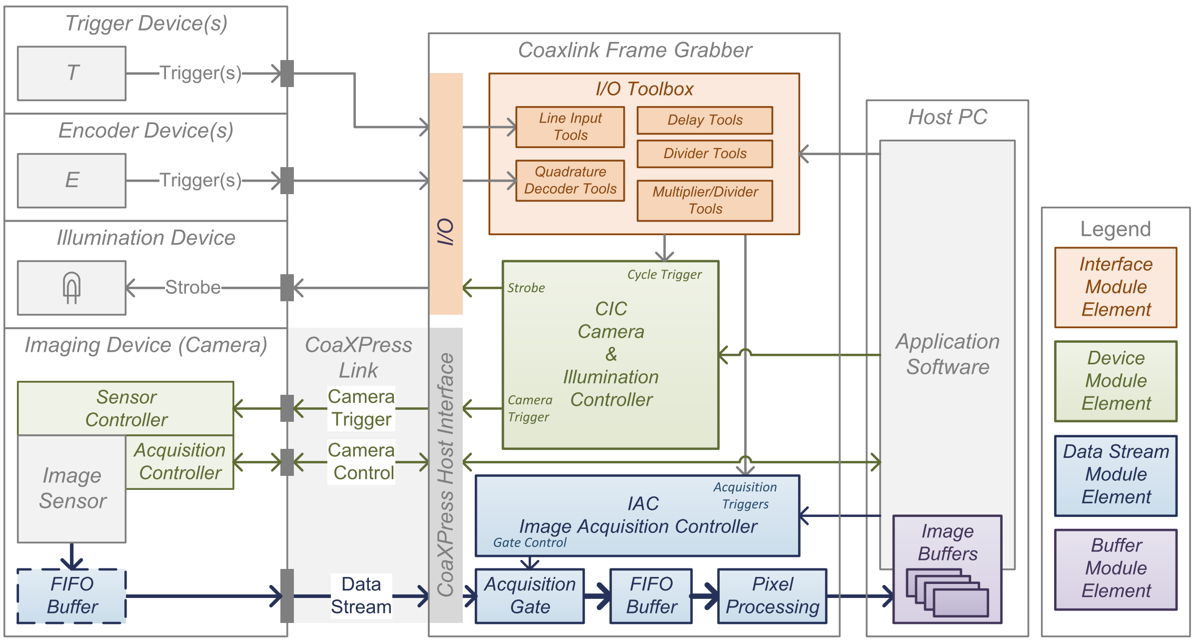
The "Image Data Transfer" on page 74 is transfers the image data to the destination buffer.

2.2. Block Diagrams

This section show the block diagram of three image acquisition systems using one Coaxlink card

> **NOTE**
> In the diagrams hereafter, the main elements are represented by rectangles and their relations are represented by line segments with arrows indicating the direction of the signal or the data flow.The filling color of the rectangle indicates the level in the GenTL hierarchy as described in the legend.

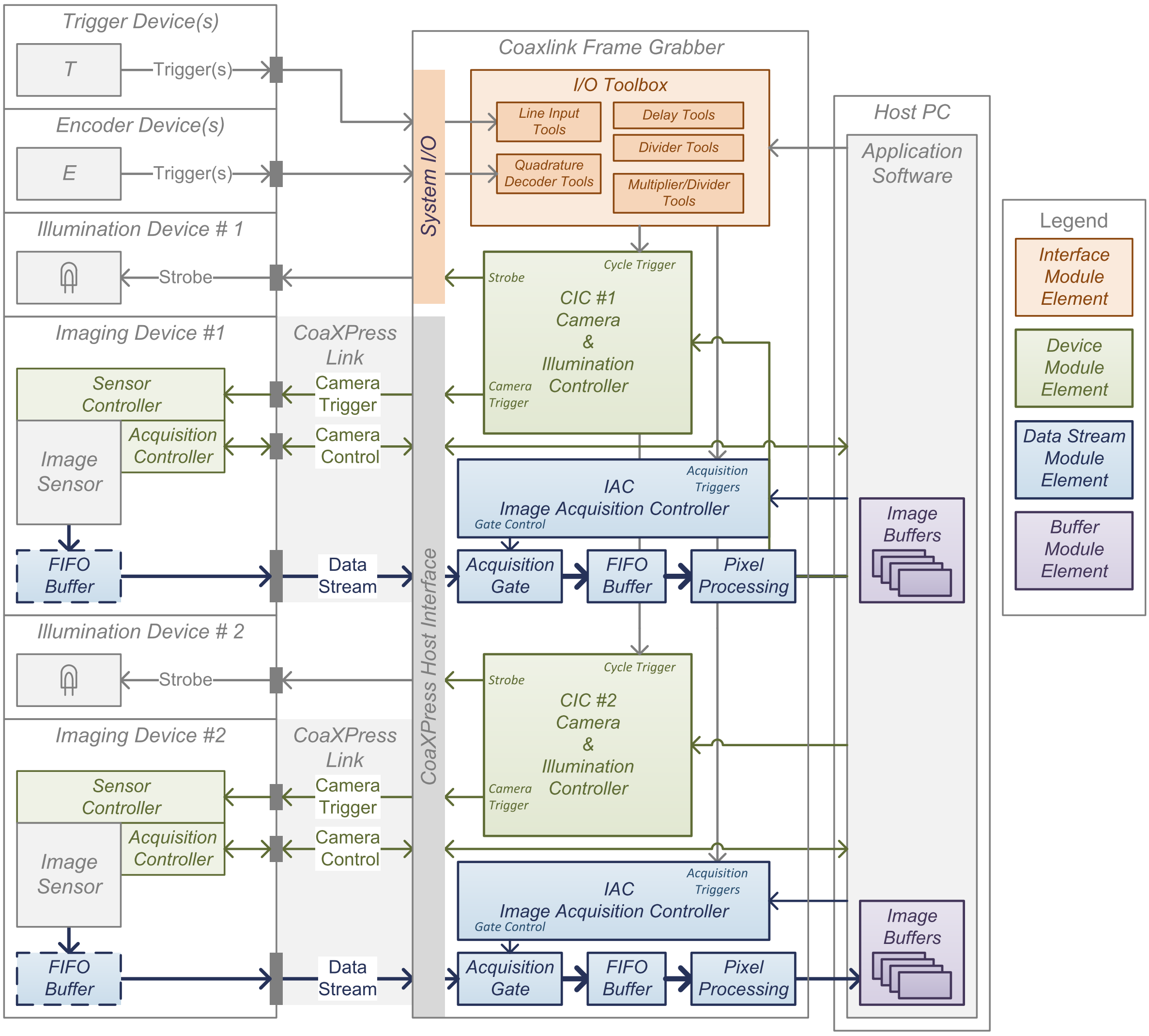
1-camera, 1-data-stream



1-camera, 1-data-stream Coaxlink image acquisition system

> **NOTE**
> this configuration applies only when a *1-camera* or a *1-camera, line-scan* firmware variant is installed.

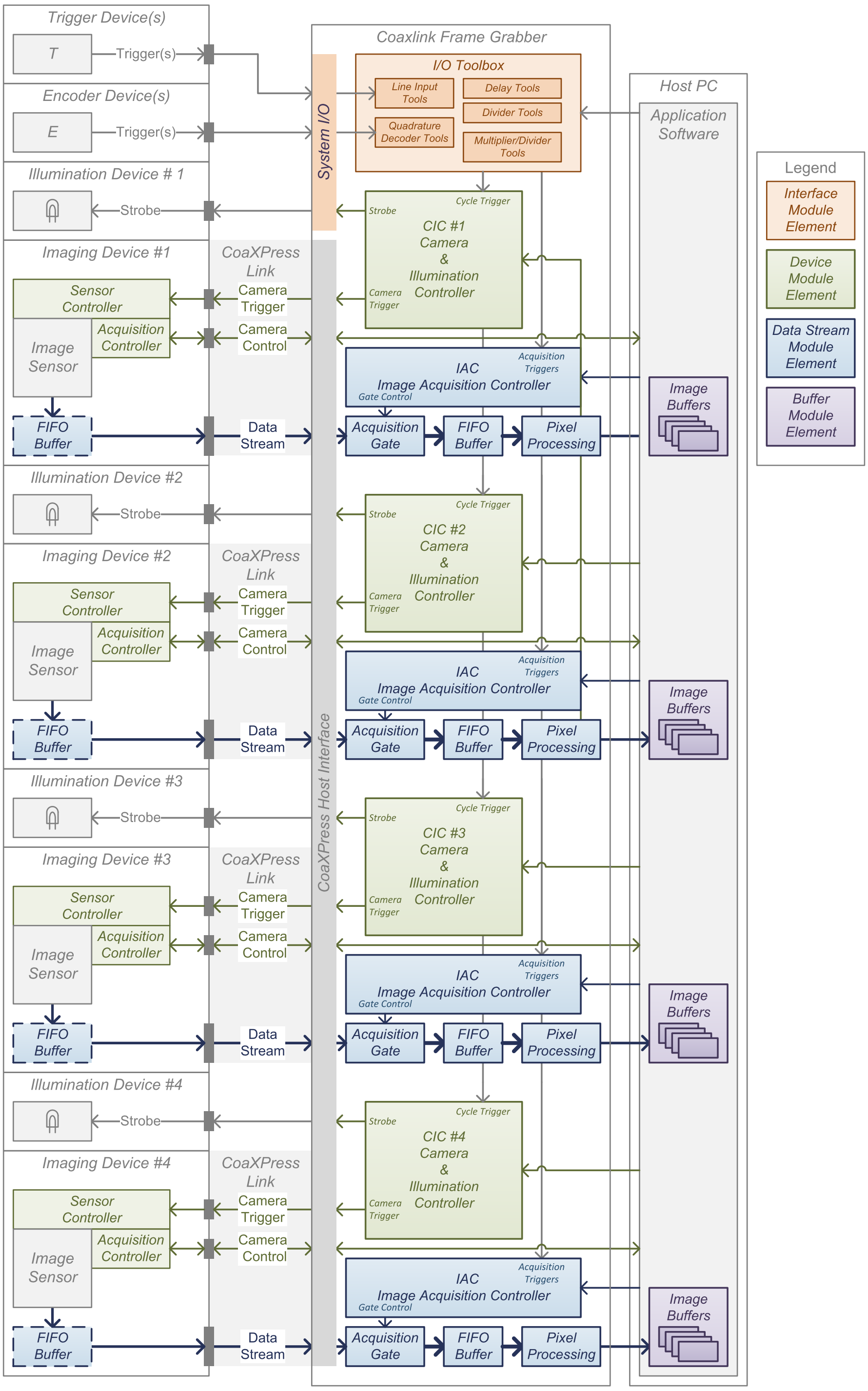
2-camera, 1-data-stream



2-camera, 1-data-stream Coaxlink image acquisition system

> **NOTE**
> this configuration applies only when a *2-camera* or a *2-camera, line-scan* firmware variant is installed.

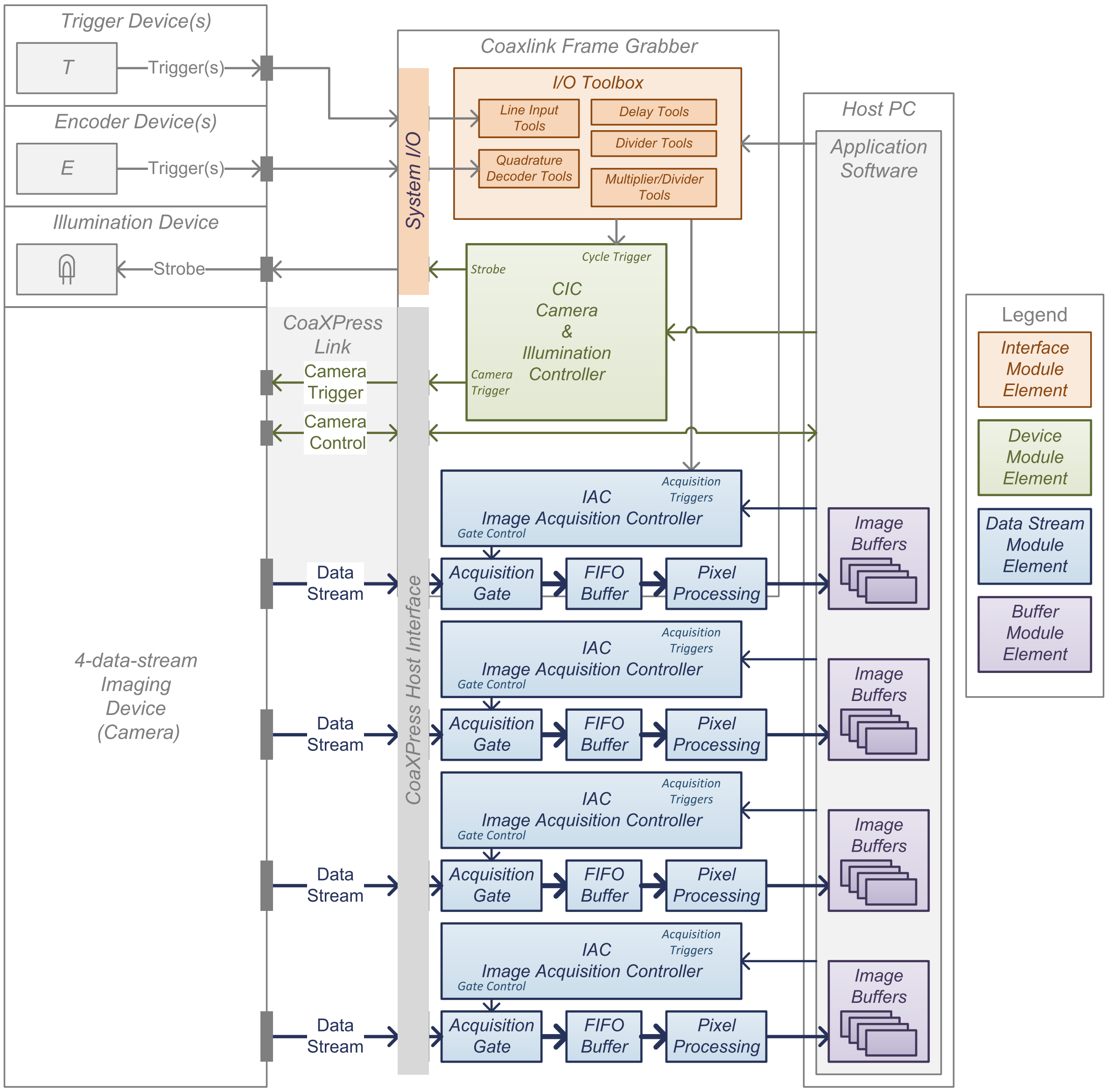
4-camera, 1-data-stream



4-camera, 1-data-stream Coaxlink image acquisition system

> **NOTE**
> this configuration applies only when a *4-camera* or a *4-camera, line-scan* firmware variant is installed.

1-camera, 4-data-stream



1-camera, 4-data-stream Coaxlink image acquisition system

> **NOTE**
> this configuration applies only when a *1-camera,4-data-stream* firmware variant is installed.

3. CoaXPress Host Interface

3.1. Host Interface Specification

Interface (Card) Specifications

Specifications having the scope of one GenTL Interface module (Namely: a Coaxlink card)

CoaXPress feature bar	Applicable products
CXP-6 DIN 1	Mono
CXP-6 DIN 2	Duo, Duo104EMB, Duo104MIL
CXP-6 DIN 4	Quad, QuadG3, QuadG3LH, QuadG3DF, Quad3DLLE
CXP-6 DIN 8	Octo
CXP-12 µBNC 1	MonoCXP12LH
CXP-12 µBNC 2	DuoCXP12
CXP-12 µBNC 4	QuadCXP12, QuadCXP12J

Maximum aggregated input data rate	Applicable products
625,000 Bytes/s	Mono
1,250,000 Bytes/s	Duo, Duo104EMB, Duo104MIL, MonoCXP12LH
2,500,000 Bytes/s	Quad, QuadG3, QuadG3LH, QuadG3DF, Quad3DLLE, DuoCXP12
5,000,000 Bytes/s	Octo, QuadCXP12, QuadCXP12J

Devices count	Applicable products
1 or 2 or 4 or 5 or 8	**See also:** "Firmware Variants per Product" on page 20

See also: CoaXPress Host Interface in the hardware manual for electrical specifications.

Device (Camera) Specifications

Specifications having of one GenTL Device module (Namely, a camera attached to a Coaxlink card)

Device Type	Applicable firmware variants
Area-scan camera, 1 data stream or Area-scan camera, up to 4 data streams or Line-scan camera, 1 data stream	**See also:** "Firmware Variants per Product" on page 20

The above table excludes the special camera types used in "CoaXPress Data Forwarding" on page 173 and "Sub-link Acquisition" on page 170 configurations.

Connections count (per camera)	Applicable products / firmware variants
1 or 2 or 4 or 8	**See also:** "Firmware Variants per Product" on page 20
8	**See also:** "Sub-link Acquisition" on page 170

Data Stream Specifications

Specifications having the scope of one GenTL Data Stream module

Maximum stream packet size	Applicable products
16,384 Bytes	All products

Image stream format	Applicable firmware variants
Rectangular Image (YSize > 0)	Area-scan "Firmware Variants per Product" on page 20
Rectangular Image (YSize = 0)	Line-scan "Firmware Variants per Product" on page 20
Arbitrary Image	*Not supported*

Image scanning method	Applicable firmware variants
Progressive	All firmware variants
Interlace	*Not supported*

Tap geometry	Applicable firmware variants
1X_1Y	All firmware variants
1X_1Y2, 1X_2YE, 1X_2YM	All area-scan firmware variants

Pixel format	Applicable firmware variants
Raw	All firmware variants
Mono8, Mono10, Mono12, Mono14, Mono16	All firmware variants
BayerGR8, BayerRG8, BayerGB8, BayerBG8 BayerGR10, BayerRG10, BayerGB10, BayerBG10 BayerGR12, BayerRG12, BayerGB12, BayerBG12 BayerGR14, BayerRG14, BayerGB14, BayerBG14 BayerGR16, BayerRG16, BayerGB16, BayerBG16	All firmware variants
RGB8, RGB10, RGB12, RGB14, RGB16	All firmware variants
RGBA8, RGBA10, RGBA12, RGBA14, RGBA16	All firmware variants

Width Increment Step

WARNING
The image width settings of the camera must be a multiple of the specified *Width Increment Step* value:

Applies to:
Mono | Duo | Quad | Duo104EMB | Duo104MIL | Quad3DLLE

Bit Depth	Width Increment Step [pixels]
8-bit	8
10-, 12-, 14- and 16-bit	*4*

Applies to:
QuadG3 | QuadG3LH | QuadG3DF

Bit Depth	Width Increment Step [pixels]	
	Bayer CFA decoder disabled	Bayer CFA decoder enabled
8-bit	4	16
10-, 12-, 14- and 16-bit	*2*	8

Applies to:
Octo | QuadCXP12 | MonoCXP12LH | DuoCXP12

Bit Depth	Width Increment Step [pixels]	
	Bayer CFA decoder disabled	Bayer CFA decoder enabled
8-bit	4	32
10-, 12-, 14- and 16-bit	*2*	16

Applies to:
QuadCXP12J

Bit Depth	Width Increment Step [pixels]
8-bit	32

3.2. Firmware Variants per Product

1630 Coaxlink Mono

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1-connection area-scan camera	1D1	LUT

1631 Coaxlink Duo

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1- or 2-connection area-scan camera	1D2	LUT
1-camera, line-scan	One 1- or 2-connection line-scan camera	1D2	LUT
2-camera	One or two 1-connection area-scan cameras	2D11	LUT
2-camera, line-scan	One or two 1-connection line-scan cameras	2D11	LUT

1632 Coaxlink Quad

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1- or 2- or 4-connection area-scan camera	1D4	LUT
1-camera, line-scan	One 1- or 2- or 4-connection line-scan camera	1D4	LUT
2-camera	One or two 1- or 2-connection area-scan cameras	2D22	LUT

1633 Coaxlink Quad G3

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1- or 2- or 4-connection area-scan camera	1D4	FFC LUT CFA
1-camera, line-scan	One 1- or 2- or 4-connection line-scan camera	1D4	FFC LUT
1-slm-camera	Master 4-connection sub-link of an 8-connection area-scan camera	1D8SLM4	LUT
1-sls-camera	Slave 4-connection sub-link of an 8-connection area-scan camera	1D8SLS4	LUT
2-camera	One or two 1- or 2-connection area-scan cameras	2D22	LUT
2-camera, line-scan	One or two 1- or 2-connection line-scan cameras	2D22	LUT
3-camera	One 1- or 2-connection and one or two 1-connection area-scan cameras	3D211	LUT
4-camera	One or two or three or four 1-connection area-scan cameras	4D1111	LUT
4-camera, line-scan	One or two or three or four 1-connection line-scan cameras	4D1111	LUT
1-camera, 4-data-stream	One 1- or 2- or 4-connection area-scan camera, up to 4 data streams	1D4S4	-

1633-LH Coaxlink Quad G3 LH

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1- or 2- or 4-connection area-scan camera	1D4	FFC LUT CFA
1-camera, line-scan	One 1- or 2- or 4-connection line-scan camera	1D4	FFC LUT
1-slm-camera	Master 4-connection sub-link of an 8-connection area-scan camera	1D8SLM4	LUT
1-sls-camera	Slave 4-connection sub-link of an 8-connection area-scan camera	1D8SLS4	LUT
2-camera	One or two 1- or 2-connection area-scan cameras	2D22	LUT
2-camera, line-scan	One or two 1- or 2-connection line-scan cameras	2D22	LUT
3-camera	One 1- or 2-connection and one or two 1-connection area-scan cameras	3D211	LUT
4-camera	One or two or three or four 1-connection area-scan cameras	4D1111	LUT
4-camera, line-scan	One or two or three or four 1-connection line-scan cameras	4D1111	LUT
1-camera, 4-data-stream	One 1- or 2- or 4-connection area-scan camera, up to 4 data streams	1D4S4	-

1629 Coaxlink Duo PCIe/104-EMB

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1- or 2-connection area-scan camera	1D2	LUT
1-camera, line-scan	One 1- or 2-connection line-scan camera	1D2	LUT
2-camera	One or two 1-connection area-scan cameras	2D11	LUT

1634 Coaxlink Duo PCIe/104-MIL

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1- or 2-connection area-scan camera	1D2	LUT
1-camera, line-scan	One 1- or 2-connection line-scan camera	1D2	LUT
2-camera	One or two 1-connection area-scan cameras	2D11	LUT

1635 Coaxlink Quad G3 DF

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1- or 2- or 4-connection area-scan camera	1D4	LUT CFA
1-df-camera	One 1- or 2- or 4-connection area-scan data-forwarded camera	1DF4	LUT CFA
1-camera, line-scan	One 1- or 2- or 4-connection line-scan camera	1D4	FFC LUT
1-df-camera, line-scan	One 1- or 2- or 4-connection line-scan data-forwarded camera	1DF4	FFC LUT

1637 Coaxlink Quad 3D-LLE

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1- or 2- or 4-connection area-scan camera	1D4	LLE

3602 Coaxlink Octo

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1- or 2- or 4- or 8-connection area-scan camera	1D8	LUT CFA
1-camera, line-scan	One 1- or 2- or 4- or 8-connection line-scan camera	1D8	LUT
2-camera	One or two 1- or 2- or 4-connection area-scan cameras	2D44	FFC LUT CFA
2-camera, line-scan	One or two 1- or 2- or 4-connection line-scan cameras	2D44	LUT
4-camera	One or two or three or four 1- or 2-connection area-scan cameras	4D2222	LUT
4-camera, line-scan	One or two or three or four 1- or 2-connection line-scan cameras	4D2222	LUT
5-camera	One 1- or 2- or 4-connection and one or two or three or four 1-connection area-scan cameras	5D41111	LUT
8-camera	Up to eight 1-connection area-scan cameras	8D11111111	LUT

3603 Coaxlink Quad CXP-12

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1- or 2- or 4-connection area-scan camera	1D4	LUT CFA
1-camera, line-scan	One 1- or 2- or 4-connection line-scan camera	1D4	LUT
2-camera	One or two 1- or 2-connection area-scan cameras	2D22	LUT
4-camera	One or two or three or four 1-connection area-scan cameras	4D1111	LUT

3620 Coaxlink Quad CXP-12 JPEG

Firmware Variant	Description	Host Connections Map	Advanced Processing
4-camera	One or two or three or four 1-connection area-scan cameras	4D1111	CFA JPEG

3621-LH Coaxlink Mono CXP-12 LH

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1-connection area-scan camera	1D1	LUT

3622 Coaxlink Duo CXP-12

Firmware Variant	Description	Host Connections Map	Advanced Processing
1-camera	One 1- or 2-connection area-scan camera	1D2	LUT
2-camera	One or two 1-connection area-scan cameras	2D11	LUT

3.3. Host Connections Maps

The CoaXPress standard suggests that Devices (cameras or data forwarding devices) can be connected to the Host (frame grabber) using a free connection scheme. Instead, the Host Interface of Coaxlink requires a specific assignment of the Device connections to the Host connectors. Such assignment is named *Host Connections Map*.

The Host Connections Map is hard-coded in the product firmware variant.

> **WARNING**
> The Coaxlink product and firmware variant must be selected according to the required mapping!

Host Connections Map naming convention

The *Host Connections Map* or *HCMAP* designates how the connections of the Host Interface of a Coaxlink card are allocated to the Devices (cameras).

A Host Connection Map - HCMAP - is designated by an acronym using the following Euresys proprietary naming convention:

<dev#><dev-type>[<str#>S]{<con#>...<con#>}[<SL-con#>]

where:

- `<dev#>`declares the maximum number of Devices (cameras) that can be attached to the Host Interface.
  - `1` for a single-device Host interface
  - `2` for a 2-device Host interface
  - ...
- `<dev-type>` declares the device type.
  - `D` for standard CoaXPress devices
  - `DF` for virtual devices used in the Data Forwarding schemes
- `<con#>` declares the number of connections available for each device.
  - 1 for a single-connection device
  - 2 for 2-connection device
  - ...

    This field is repeated once for each device.
- <str#>S declares the maximum number of data streams allowed by a device.
  - This field is omitted when there is only 1 stream
  - 4S for a up to 4 data-streams per device
- `SL<-con#>` declares the number connections per sub-link.
  - `SL4` for a 4-connection sub-links

    This field is omitted when there are no sub-links.

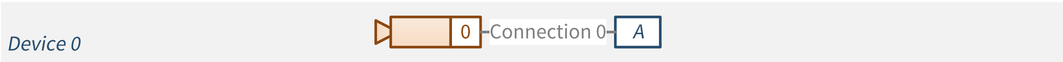
Examples

HCMAP *2D22* designates a Host Interface with 2 standard 1-data-stream CoaXPress Devices and 2 connections for each device.

HCMAP *1D4S4* designates a Host Interface with 1 standard CoaXPress Devices, up to 4 data streams, and 4 connections per device.

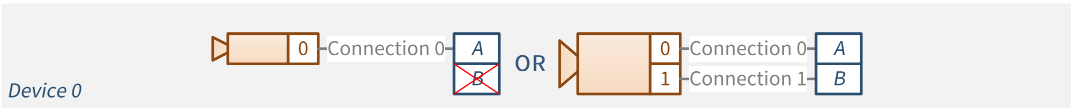
1D1 host connections map

One 1-connection device



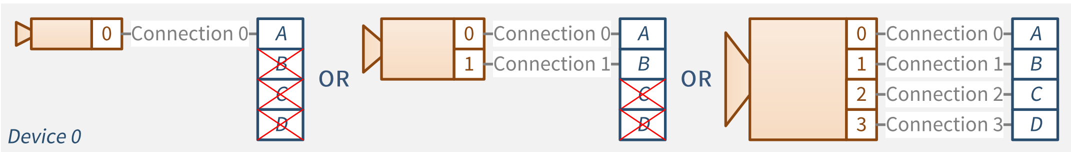
1D2 host connections map

One 1- or 2-connection device



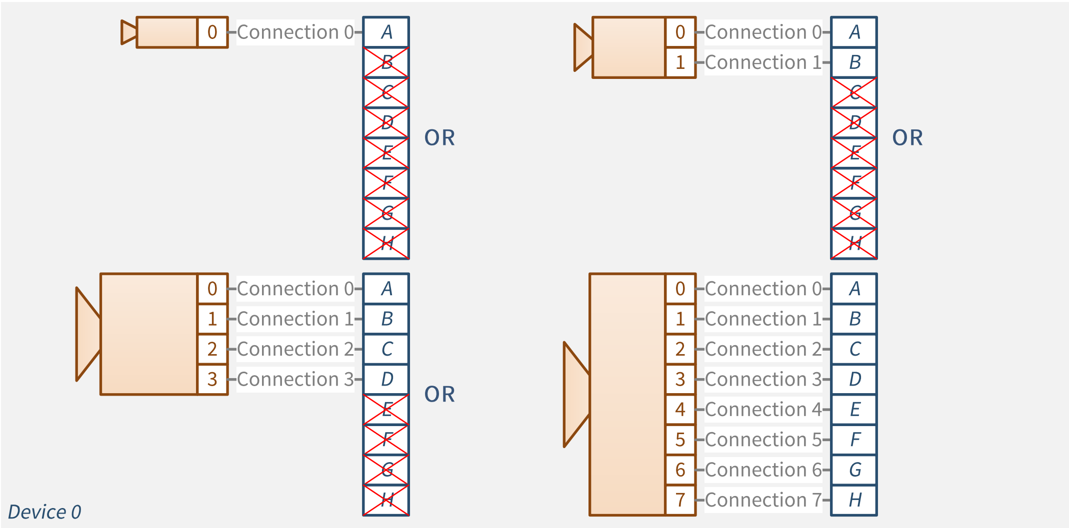
1D4 host connections map

One 1- or 2- or 4-connection device



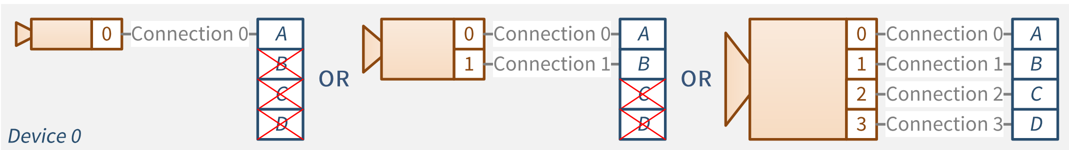
1D8 host connections map

One 1- or 2- or 4- or 8-connection device



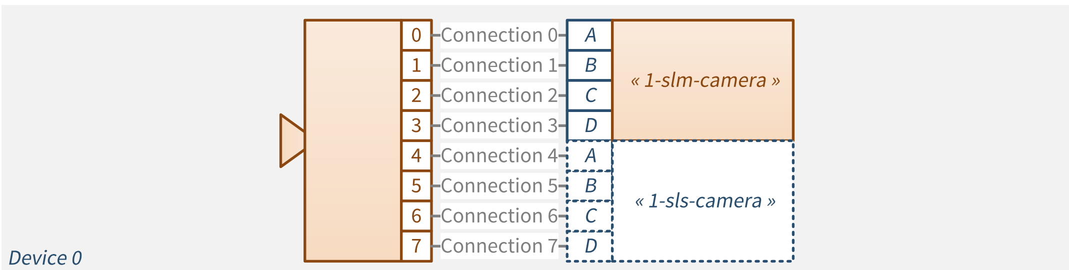
1D4S4 host connections map

One 1- or 2- or 4-connection device, up to 4 data streams



1D8SLM4 host connections map

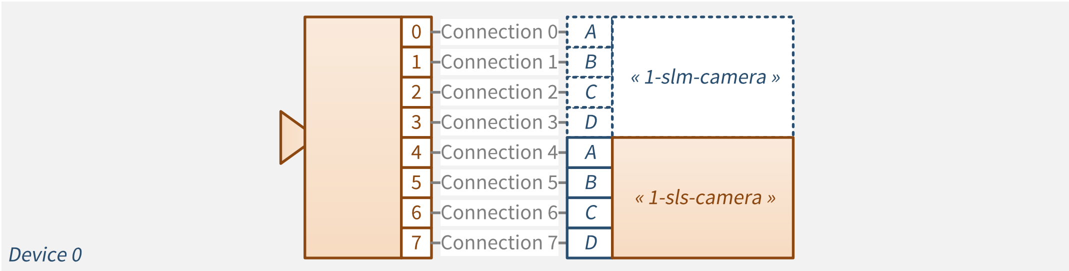
Master 4-connection sub-link of an 8-connection device



See also: "Sub-link Acquisition" on page 170 for the connection scheme of an 8-connection camera to two Coaxlink cards.

1D8SLS4 host connections map

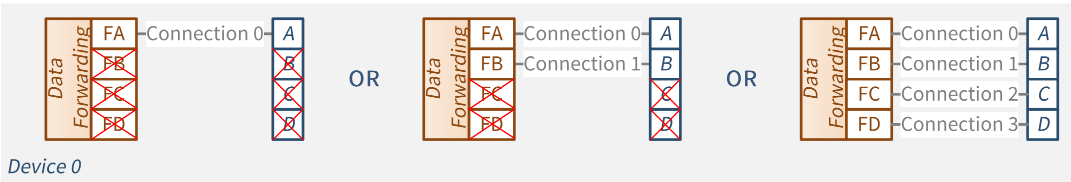
Slave 4-connection sub-link of an 8-connection device



See also: "Sub-link Acquisition" on page 170 for the connection scheme of an 8-connection camera to two Coaxlink cards.

1DF4 host connections map

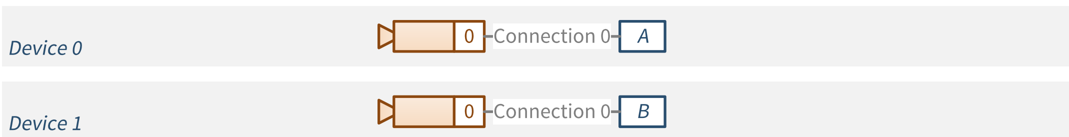
One 1- or 2- or 4-connection device



See also: "CoaXPress Data Forwarding" on page 173 for the connection schemes of slave Data Forwarding devices.

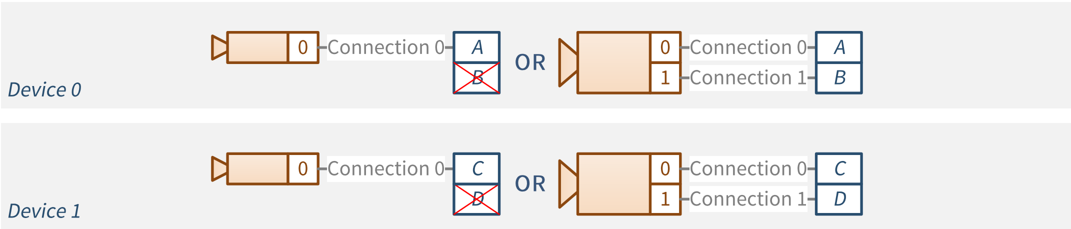
2D11 host connections map

One or two 1-connection devices



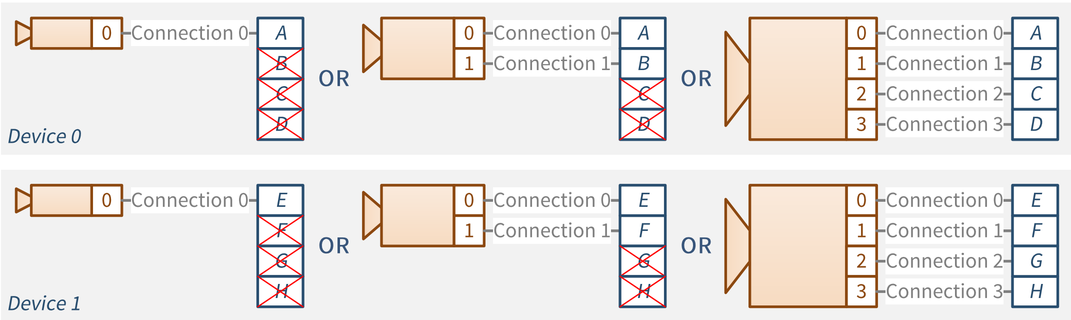
2D22 host connections map

One or two 1- or 2-connection devices



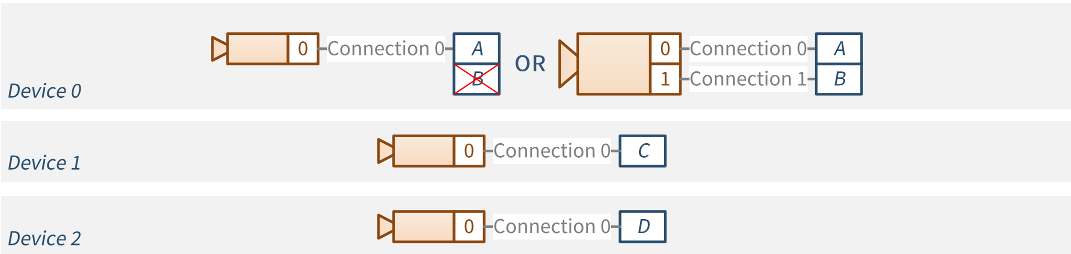
2D44 host connections map

One or two 1- or 2- or 4-connection devices



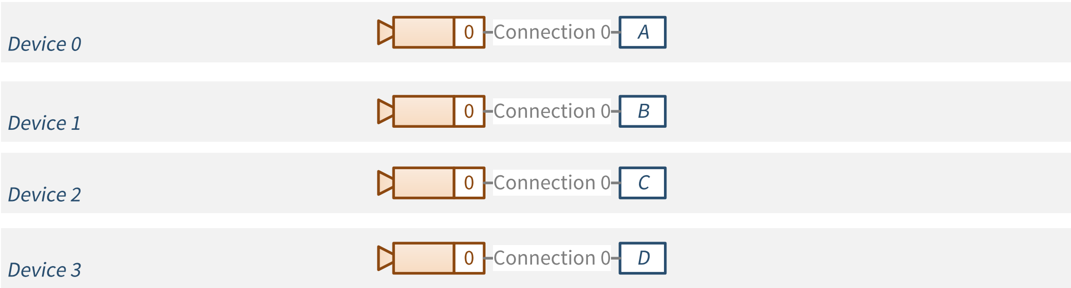
3D211 host connections map

One 1- or 2-connection and one or two 1-connection devices



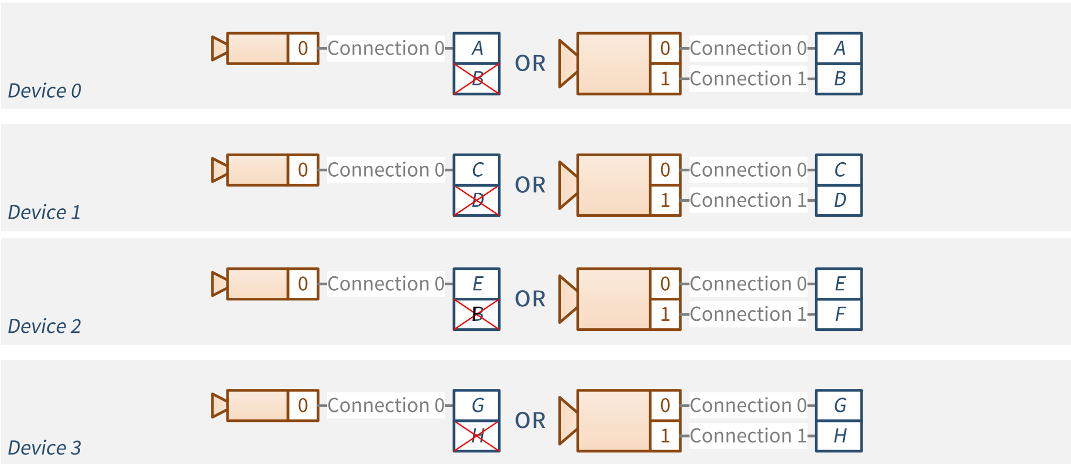
4D1111 host connections map

One or two or three or four 1-connection devices



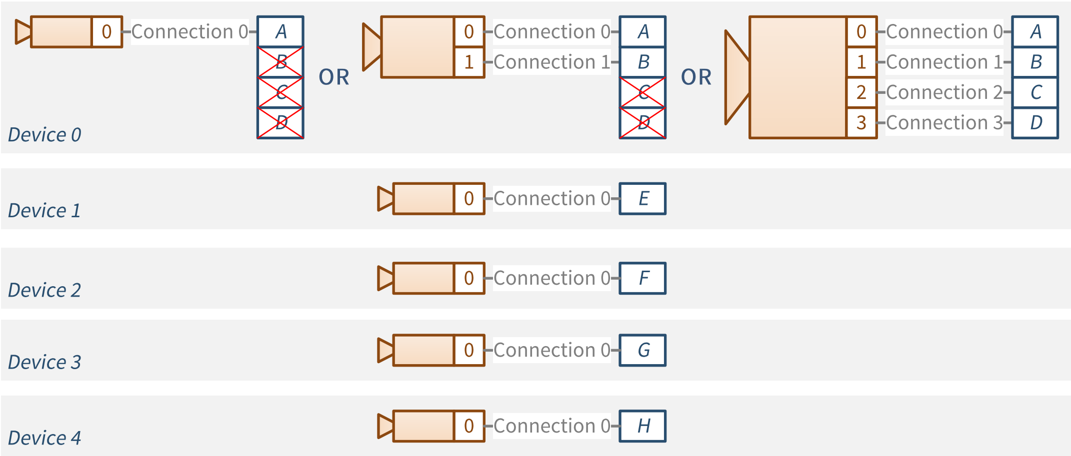
4D2222 host connections map

One or two or three or four 1- or 2-connection devices



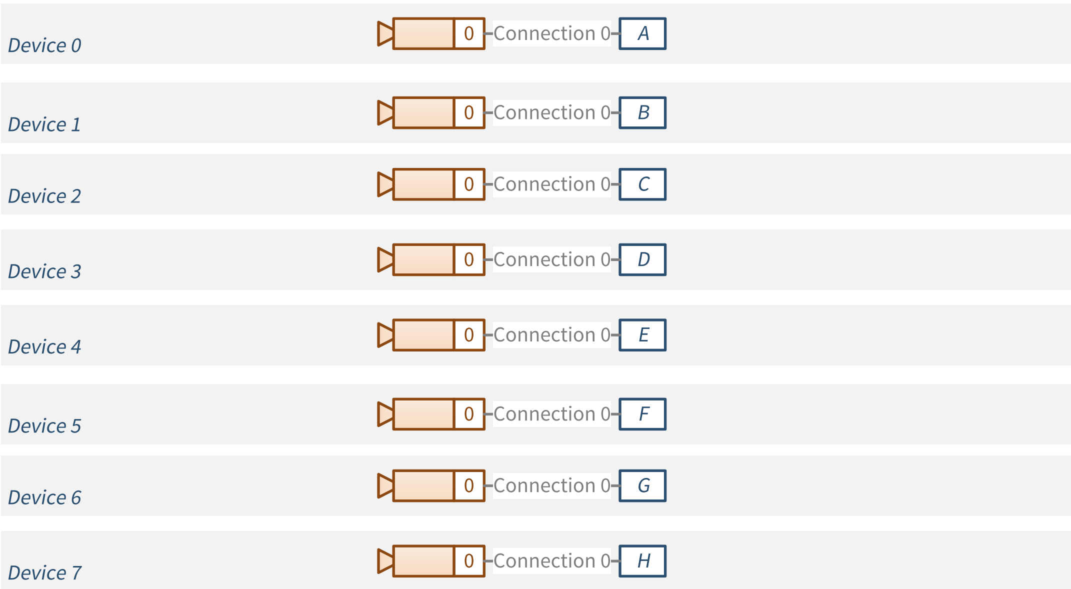
5D41111 host connections map

One 1- or 2- or 4-connection and one or two or three or four 1-connection devices



8D11111111 host connections map

Up to eight 1-connection devices



3.4. Link Configuration

Automatic Link Configuration

The Coaxlink driver provides an automatic link discovery and configuration for CoaXPress 1.0 and CoaXPress 1.1 devices.

For each connection of the CoaXPress Host interface, the discovery procedure determines:

- The presence of a CoaXPress Device
- The speed of the down-connection (Device to Host)
- The connection ID

The discovery results are reported through the CxpConnectionState, CxpDownConnectionSpeed and CxpDeviceConnectionID features of the Interface module.

The user is invited to check if the resulting link configuration is appropriate:

- For the application needs in terms of link bandwidth (link speed and number of connections)
- For Coaxlink in terms of camera connection schemes supported by the target Coaxlink product/firmware combination

Manual Link Configuration

If necessary, the user can manually configure the CoaXPress link of the Remote Device.

This can be achieved, regardless of the camera brand, by assigning the appropriate value to the CxpLinkConfiguration GenApi feature of the Coaxlink device module.

Assigning the value Preferred enforces the preferred link configuration of the camera:

- The link speed is set to the specified value
- The link width is set to the specified value but, possibly limited to the number of available connections on the Coaxlink side

3.5. Power Over CoaXPress

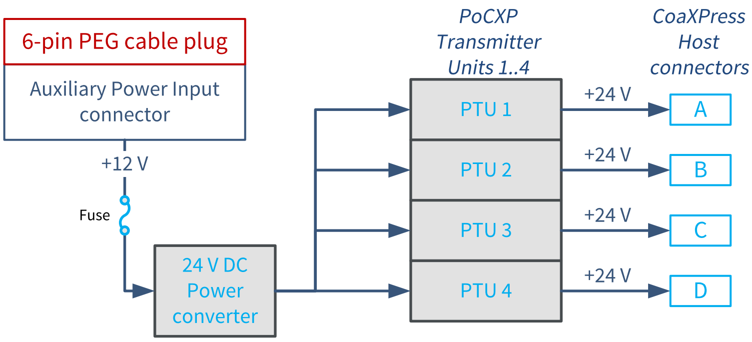
Each connection of the CoaXPress Host connector is capable of delivering power to the camera through the CoaXPress cable.

Power sources

12 V DC power source

Applies to:

Mono	Duo	Quad	QuadG3	QuadG3LH	QuadG3DF
Quad3DLLE	Octo	QuadCXP12	QuadCXP12J	MonoCXP12LH	DuoCXP12



PoCXP power distribution scheme for cards with 12 V DC power source

An external 12 V DC supply is attached to the *auxiliary power input connector* through a 6-pin PEG cable. The 12 V power is converted by a 12 V to 24 V regulated converter and delivered to each power transmitting unit.

NOTE
The 12 V DC supply is typically delivered by the power supply of the Host PC.

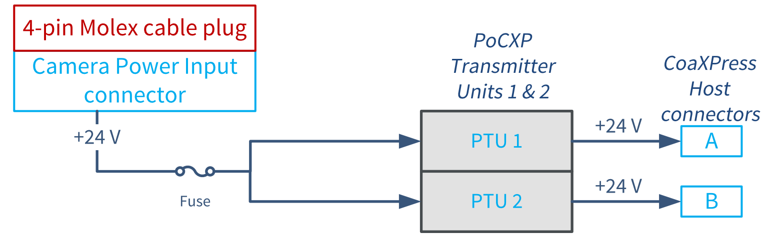
The AuxiliaryPowerInput feature reports the status of the connection made by the PEG cable between the external power supply and the Coaxlink auxiliary power input connector.

The CxpPoCxpPowerInputStatus feature reports the status of the 24 V power converter.

24 V DC power source

Applies to:

Duo104EMB Duo104MIL



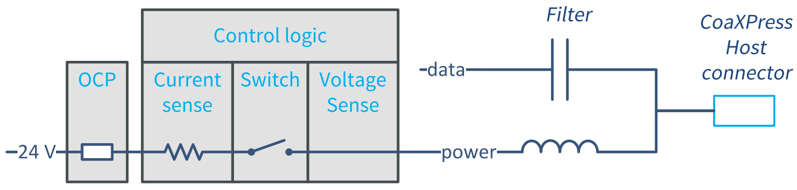
PoCXP power distribution scheme for cards with 24 V DC power source

The external 24 V supply is attached to the *auxiliary power input connector* through a 4-pin cable and delivered to each power transmitting unit.

The AuxiliaryPowerInput feature reports the status of the connection made by the power cable between the external power supply and the Coaxlink auxiliary power input connector.

The CxpPoCxpPowerInputStatus feature reports the status of the 24 V input.

Power transmitter unit



PoCXP power transmission unit

The Power Transmitting Unit – PTU – is responsible for a safe delivery of power to the Device. It fulfills the requirements of the CoaXPress standard for a CoaXPress Host, namely:

- It is capable of delivering 17 W* of 24 V DC power per connector to the Device
  (*) 25 W for **3621-LH Coaxlink Mono CXP-12 LH**
- It implements an over-current protection device – OCP
- It supports the automatic CoaXPress PoCXP detection method

In addition, it provides the application with the capability of:

- Disabling or interrupting the automatic power delivery
- Resetting the OCP when tripped
- Controlling the range of the PoCXP sense resistance

Automatic PoCXP control

On execution of the CxpPoCxpAuto command the PTU controller initiates a *PoCXP device detection procedure*.

> **NOTE**
> Since version 3.1 of the Coaxlink Driver, the automatic PoCXP is enabled at system power-up! The application is not anymore required to enable PoCXP powering by issuing a `CxpPoCxpAuto` command.

If the PoCXP device detection procedure terminates successfully, the PTU applies power by closing the switch.

If the *PoCXP device detection procedure fails*, the controller doesn't apply power and retries a new PoCXP detection procedure. Possible causes of failure are:

- The external power is not connected (AuxiliaryPowerInput = Unconnected)
- The external power source is off (CxpPoCxpPowerInputStatus = NotOK)
- There are no camera attached
- The attached camera is not PoCXP compliant

Once the power is applied, the controller remains in that state until any of the following situations occurs:

- The application disables the power delivery by executing the PoCxpTurnOff command.
- The external power source is disconnected (CxpPoCxpPowerInputStatus = NotOK)
- The external power source is turned off (CxpPoCxpPowerInputStatus = NotOK)
- The CoaXPress cable is disconnected (The average output current measured over a time interval of 0.3 seconds is less than 8 mA)
- The OCP trips (The output voltage drops for a time interval greater than 20 milliseconds)

Manual PoCXP control

On execution of the CxpPoCxpTurnOff command the PTU turns off the switch and disables PoCXP powering. In that state, the PTU is not performing PoCXP detection procedures.

The CxpPoCxpConfigurationStatus feature reports the configuration status of the PTU: Off or AUTO.

The CxpPoCxpStatus feature reports the status of the PTU: Off, On or Tripped.

PoCXP detection mode control

The CxpPoCxpDetectionMode feature of the Interface module selects either the standard or the extended (default) power over CoaXPress detection mode.

When set to Extended, the PoCXP device detection of Coaxlink cards is configured for an extended range of resistance values. This allows cameras that are not fully compliant with the range specification of the PoCXP sense resistor to be detected as valid PoCXP cameras and to be powered. This is the default value after initialization.

When set toStandard the PoCXP device detection of Coaxlink cards is configured for a restricted range of resistance values, namely 4.7 kΩ +/- 10%.

WARNING
This setting is not persistent.

Over-current protection

The OCP circuit is built with a PTC device providing two kind of protections:

- The overload protection addresses the cases when the load is excessive.
- The short-circuit protection addresses the cases of accidental short-circuits.

In case of overload, the PTC trips (= opens progressively the circuit) after several seconds or minutes depending on the current level and the ambient temperature. The higher the current, the lower the time to trip. The same applies to the ambient temperature.

In case of short-circuit, the PTC trips immediately. Consequently, the PTU controller enters the tripped state and opens the switch. The tripped PTC device returns to the conducting state after having cooled down. This may take a few seconds. However, the PTU controller remains in the tripped state until the application issues a CxpPoCxpTripReset command. Having left the tripped state, the PTU can initiate a new PoCXP device detection and, if successful, re-establish power.

NOTE
The PTC is sized to sustain 17 W* of power over the whole operating temperature range without tripping.
(*) 25 W for **3621-LH Coaxlink Mono CXP-12 LH**

NOTE
Extracting more than 17 W* of power and/or operating the Coaxlink card above the operating temperature range is prohibited since it may induce unexpected PTC trips.
(*) 25 W for **3621-LH Coaxlink Mono CXP-12 LH**

3.6. CoaXPress I/O Channel

According to the CoaXPress 1.0 and 1.1 standards, the CoaXPress I/O Channel:

- Is one of the three logical channels of the CoaXPress Link (I/O, Stream, Control)
- Is defined only for the master connection (connection 0) of a CoaXPress Link
- Is used for transmitting of high-priority "triggers" between the Host and the Device
- On CoaXPress 1.0 only, is used for exchanging the state of GPIO registers between the Host and the Device

Coaxlink implements only the CoaXPress Host to Device trigger!

NOTE
CoaXPress Device to Host trigger and CoaXPress 1.0 GPIO are NOT implemented.

3.7. CoaXPress Host To Device Trigger

The CoaXPress Host To Device Trigger is a functionality of the CoaXPress I/O Channel that allows the Host (frame grabber) to trigger the Device (camera) through the CoaXPress Link.

The CoaXPress Host Interface of Coaxlink implements one CoaXPress Host to Device trigger transmitter for *each connected Device*.

Host to Device Trigger Source

The CoaXPress Host to Device Trigger transmitter can be sourced from:

- The Camera Trigger output of the associated Camera and Illumination Controller
- Any input-capable General Purpose I/O

The trigger source is indirectly controlled through the CameraControlMethod GenICam feature.

- When CameraControlMethod is set to RG or RC, the trigger source is the Camera Trigger output of the associated Camera and Illumination Controller.
- When CameraControlMethod is set to EXTERNAL:
  - The trigger source is the line source of a dedicated LIN tool of the I/O Toolbox: LIN1 for Device0, LIN2 for Device1, LIN3 for Device2, and LIN4 for Device3.
  - Any input-capable GPIO line can be used as trigger source by configuring the line source – LineInputToolSource – of the dedicated LIN tool.
  - The polarity of the external trigger signal can be controlled with the LineInverter setting of the selected I/O Control block.
  - The time constant of the glitch-removal filter can be adjusted through the LineFilterStrength setting of the selected I/O Control block.
- When CameraControlMethod is set to NC, the Host to Device Trigger transmitter is disabled.

Host to Device Trigger Transmitter - Default Settings

The Coaxlink implementation of the CoaXPress Host to Device Trigger transmitter complies with the requirements of the CoaXPress 1.0 and 1.1 standards for a low-speed CoaXPress Host to Device Trigger when it is configured with the default settings:

- CxpTriggerMessageFormat = Pulse
- CxpTriggerAckTimeout = 20.0
- CxpTriggerMaxResendCount = 3

The transmitter *initiates* a *trigger transaction* on both edges of the trigger source signal:

- It computes a delay value allowing the receiving device to recreate the event with a fixed latency.

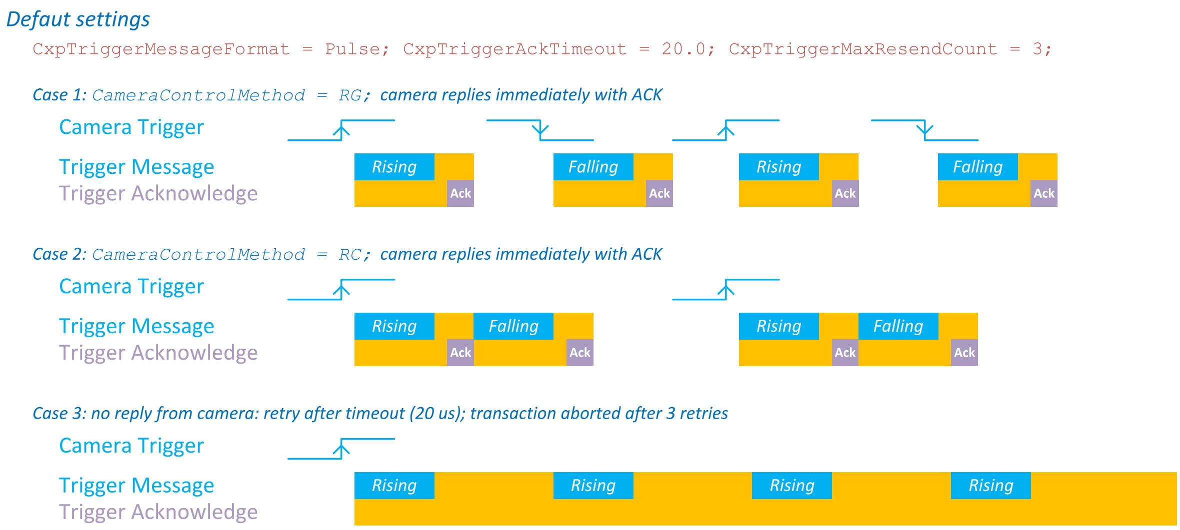
See also: "Camera trigger latency compensation" on page 46

- It inserts a high-priority "trigger packet" on the low-speed host-to-device connection at the next character boundary.

Then, the transmitter waits for the *acknowledgment* from the Device (camera):

- If the acknowledgment is received before the expiration of the timeout, the transaction terminates normally.
- If no acknowledgment is received within the 20 µs timeout, the transmitter performs a *retry*: it resends the trigger packet and initiates a new waiting period for the acknowledgment.
- If no acknowledgment is received after 3 times, the transaction terminates abnormally.

The transmitter doesn't initiate a new transaction while the previous one is not completed.



Trigger message transactions using default settings
Case 1 and case 2: the camera acknowledges each message as expected
Case 3: no acknowledgment from camera. Abort after 3 retries

Events Reporting

The transmitter reports the following events:

- CxpTriggerAck: Received acknowledgment for CoaXPress Host to Device trigger packet.
- CxpTriggerResend: Resent CoaXPress Host to Device trigger packet.

Host to Device Trigger Transmitter - Alternate Settings

The transmitter can be customized:

- To send trigger messages only on the rising edge of the source signal using the *Message Format Control*
- To configure the acknowledge timeout and the number of retries using the *Message Acknowledge Control*

Message Format Control

The Host to Device Trigger transmitter unit of Coaxlink provides a "message format" control with the CxpTriggerMessageFormat GenICam feature.

Pulse Message Format (Default)

By default, CxpTriggerMessageFormat is set to Pulse: the transmitter generates a CoaXPress I/O Channel Host to Device Trigger transaction on *both edges* of the input pulse:

- The transaction initiated by the rising edge transmits a *rising edge trigger packet* from the Host to the Device.
- The transaction initiated by the falling edge transmits a *falling edge trigger packet* from the Host to the Device.

> **NOTE**
> Every trigger pulse requires two distinct CoaXPress I/O Channel transactions!

Rising Edge Message Format

When CxpTriggerMessageFormat is set to RisingEdge, the transmitter generates a CoaXPress I/O Channel Host to Device Trigger transaction on *the rising edge only* of the input pulse.

The transaction always transmits a *rising edge trigger packet* from the Host to the Device.

> **NOTE**
> Every trigger pulse requires a single CoaXPress I/O Channel transaction.

> **NOTE**
> This format does not allow the grabber to control the exposure time!

Toggle Message Format

When CxpTriggerMessageFormat is set to Toggle, the transmitter generates a CoaXPress I/O Channel Host to Device Trigger transaction on *the rising edge only* of the input pulse.

The transaction alternatively transmits a *rising edge trigger packet* and a *falling edge trigger packet*.

> **NOTE**
> Every trigger pulse requires a single CoaXPress I/O Channel transaction.

> **NOTE**
> This format does not allow the grabber to control the exposure time!

The CxpTriggerLevel feature allows the application to set and/or get the current level of the CoaXPress Host to Device Trigger signal.

Message Acknowledge Control

The Host to Device Trigger transmitter unit provides a user-configurable trigger packet acknowledgment mechanism:

- The time-out value is configurable using the CxpTriggerAckTimeout GenICam feature.
- The number of retries is configurable using the CxpTriggerMaxResendCount GenICam feature.

Enable Acknowledge Checking (Default)

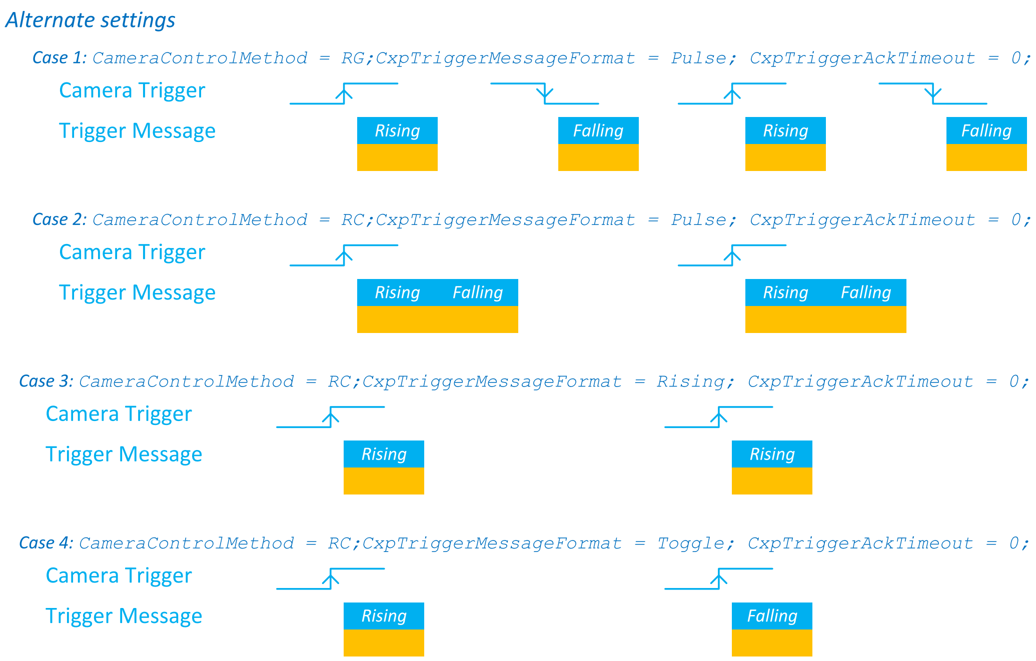
By default, CxpTriggerAckTimeout is set to 20.0 (20 microseconds) and CxpTriggerMaxResendCount is set to 3.

Coaxlink expects an I/O Channel Acknowledgment packet in response to every Trigger packet. If the acknowledgment packet is not received within the 20 µs time-out value, the transmitter resends the trigger packet. It performs up to 3 retries.

Setting larger CxpTriggerAckTimeout values allows more time for the Device to acknowledge the trigger packet.

Disable Acknowledge Checking

Setting CxpTriggerAckTimeout to 0 disables the acknowledgement mechanism. The trigger transaction terminates immediately after having sent the trigger packet.



Trigger message transactions using alternate settings to allow higher trigger rates

Alternate settings for fastest trigger rate

The fastest trigger rate of : *595.2 kHz @CXP-10 and CXP-12 link speeds* or *297.6 kHz @CXP-6 and lower link speeds* can be achieved when:

- CameraControlMethod = RC (asynchronous reset camera, camera-controlled exposure),
- CxpTriggerAckTimeout = 0 (acknowledge ckecking disabled),
- CxpTriggerMessageFormat = Rising or CxpTriggerMessageFormat = Toggle.

Camera trigger latency compensation

Trigger accuracy

The Host to Device trigger packets are transmitted over the low-speed up-connection of the CoaXPress Link. The transmission of trigger packets can only start at the boundary of a character. This introduces a jitter corresponding to one character transmission time:

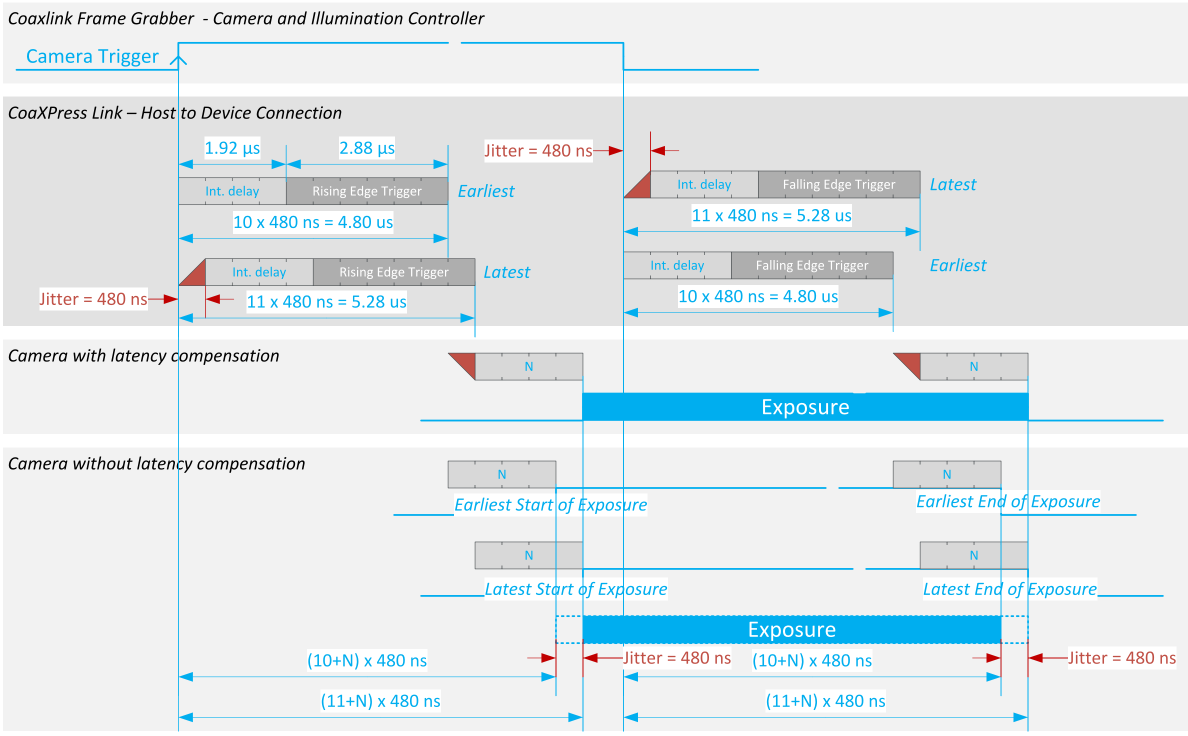
- *240 nanoseconds* @CXP-10 and CXP-12 link speeds or
- *480 nanoseconds* @CXP-6 and lower link speeds.

To minimize trigger jitter, the time between the trigger event and the trigger packet being sent is encoded into the trigger packet as a delay value expressed in units of 1/24th of the bit period:

- *1 nanosecond* @CXP-10 and CXP-12 link speeds or
- *2 nanoseconds* @CXP-6 and lower link speeds.

The receiver (camera) can then use this value to recreate the trigger event with a *fixed latency*. It compensates the transmission jitter by delaying the decoded message by the remaining fraction of one character time.

CoaXPress Camera Trigger transmission timing @CXP-6 and lower link speeds



CIC Camera Trigger to Sensor Exposure timing diagrams @CXP-6 and lower link speeds

The above diagram shows the time delay required to propagate Camera Trigger events from the frame grabber up to the camera through the CoaXPress Link using Host to Device CoaXPress Trigger messages.

The above diagram assumes that:

- CameraControlMethod is set to RG.
- The camera properly acknowledges the trigger messages and effectively initiates a new exposure.

The delay from the rising (or the falling) edge of the Camera Trigger signal (inside the Coaxlink card) up the CoaXPress link is composed of:

- A variable delay of *0-480 ns* corresponding to the time delay until the next character boundary on the low-speed CoaXPress connection.
- A fixed delay of *1.92 µs* corresponding to a 4-character pipeline delay in the Trigger Transmitter implementation.
- A fixed delay of *2.88 µs* corresponding to a 6-character message transmission time.

The delay from the CoaXPress Link to the effective start (or end) of exposure is camera-dependent. In the above drawing, this delay is assumed to be N character times (with N=4).

Jitter-compensated cameras

When the camera implements the CoaXPress jitter compensation, the one-character jitter (480 ns) introduced by the transmitter can be entirely compensated.

The overall latency is *fixed* but it remains camera-dependent: the lowest possible latency with Coaxlink products is *11 x 480 ns*, i.e. *5.28 µs*.

The residual jitter after compensation can be as low as *4 ns*.

Jitter-Uncompensated cameras

When the camera doesn't implement the CoaXPress jitter compensation, the one-character jitter (*480 ns*) introduced by the transmitter remains.

The overall latency is *variable* and camera-dependent: the lowest possible latency with Coaxlink products is *(10 ~ 11) x 480 ns*, i.e. *(4.80 ~ 5.28) µs*.

CoaXPress Camera Trigger transmission timing @CXP-10 and CXP-12 link speeds

The delay from the rising (or the falling) edge of the Camera Trigger signal (inside the Coaxlink card) up the CoaXPress link is composed of:

- A variable delay of *0-2400 ns* corresponding to the time delay until the next character boundary on the low-speed CoaXPress connection.
- A fixed delay of *0.96 µs* corresponding to a 4-character pipeline delay in the Trigger Transmitter implementation.
- A fixed delay of *1.44µs* corresponding to a 6-character message transmission time.

The delay from the CoaXPress Link to the effective start (or end) of exposure is camera-dependent. In the above drawing, this delay is assumed to be N character times (with N=4).

Jitter-compensated cameras

When the camera implements the CoaXPress jitter compensation, the one-character jitter (480 ns) introduced by the transmitter can be entirely compensated.

The overall latency is *fixed* but it remains camera-dependent: the lowest possible latency with Coaxlink products is *11 x 240 ns*, i.e. *2.64 µs*.

The residual jitter after compensation can be as low as *2 ns*.

Jitter-Uncompensated cameras

When the camera doesn't implement the CoaXPress jitter compensation, the one-character jitter (*240 ns*) introduced by the transmitter remains.

The overall latency is *variable* and camera-dependent: the lowest possible latency with Coaxlink products is *(10 ~ 11) x 240 ns*, i.e. *(2.40 ~ 2.64) µs*.

3.8. CoaXPress LED lamps

Each connector of the CoaXPress Host Interface is associated with a LED lamp mounted on the bracket (for PCie cards only).

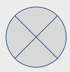
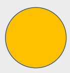
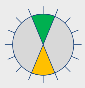
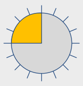
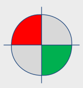
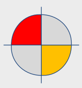
LED lamps mode control

The LampMode feature of the Interface module defines the lamps operation mode:

- When set to Standard (default value), the lamps indicate the state of the CoaXPress Link connection.
- When set to Dark, all lamps are turned off.
- When set to Error, all lamps are turned off unless error conditions are detected.
- When set to Custom, all lamps are controlled by LampCustomValue, a bitfield where each bit is mapped onto a lamp with 1 for orange and 0 for off by the LampCustomLedA … LampCustomLedH boolean features.

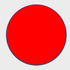
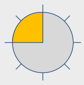
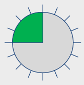
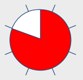
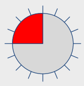
CoaXPress Host Indicator LED lamps states

States description

Symbol	Indication	State
	Off	No power
	Solid orange	System booting
	AlternateFlash_12_5 green / orange[1]	Connection detection in progress; PoCXP active
	Flash_12_5 orange[2]	Connection detection in progress; PoCXP not in use
	AlternateFlash_0_5 red / green	Device/ Host incompatible; PoCXP active
	AlternateFlash_0_5 red / orange	Device/ Host incompatible; PoCXP not in use

[1]Shown for a minimum of 1 second even if the connection detection is faster

[2]Shown for a minimum of 1 second even if the connection detection is faster

Symbol	Indication	State
	Solid red	PoCXP over-current
	Solid green	Device / Host connected, but no data being transferred
	Flash_1 orange	Device / Host connected, waiting for event (e.g. trigger, exposure pulse)
	Flash_12_5 green	Device / Host connected, data being transferred
	500 ms red pulse[1]	Error during data transfer (e.g. CRC error, single bit error detected)
	AlternateFlash_0_5 green / orange	Connection test packets being sent
	Flash_12_5 red	System error (e.g. internal error)

Flashing states timing definitions

Indication	Frequency	Duty Cycle
Flash_12_5	12.5 Hz	25% (20 milliseconds on, 60 milliseconds off)
Flash_1	1 Hz	20% (200 milliseconds on, 800 milliseconds off)
Flash_0_5	0.5 Hz	50% (1 second on, 1 second off)
AlternateFlash_12_5	12.5 Hz	25% (20 milliseconds on color 1, 60 milliseconds off, 20 milliseconds on color 2, 60 milliseconds off)
AlternateFlash_0_5	0.5 Hz	50% (1 second on color 1, 1 second off, 1 second on color 2, 1 second off)

[1] In case of multiple errors, there shall be at least two green Flash_12_5 pulses before the next error is indicated

3.9. Connection Test

The CoaXPress Host Interface of Coaxlink provides connection test facilities to test the quality up- and down-connections of the CoaXPress link according to the procedures defined in section 8.7 of the CoaXPress 1.1 standard.

For each individual CoaXPress connector, it implements

- A test generator
- A test receiver

The test generator transmits a Test Data Packet containing a known test pattern produced by a sequence generator. It increments the packet counter for each test packet transmitted.

The test receiver compares the received test data packet content against its local sequence generator. It increments the error counter for each word that is different in the data packet, and increments the packet counter for each test packet received.

NOTE
The test packet counters show how many test packets have been sent and received, so allowing a judgment to be made on the statistical meaning of the value in the error counter.

NOTE
Both Device to Host and Host to Device connection tests can be run at the same time.

3.10. CoaXPress 2.0 Error Counters

The "CoaXPress 2.0 error counters" keep track of errors that the CoaXPress protocol can detect on each individual CoaXPress connection.

Error counters

Connection lock loss counters

There is one counter per CoaXPress host connector instance that counts the number of lock losses encountered by the CoaXPress receiver.

8b/10b encoding error counters

There is one counter per CoaXPress host connector instance that counts the number of 8b/10b encoding errors encountered by the CoaXPress receiver.

Duplicated characters mismatch counters

There are two counters per CoaXPress host connector instance that counts the number of duplicated characters mismatch encountered by the CoaXPress receiver:

- The first counter counts the occurrences that could be corrected.
- The second counter counts the occurrences that could NOT be corrected.

CRC error counters

There are three counters per CoaXPress host connector instance that counts the number of CRC errors encountered by the CoaXPress receiver:

- The first counter counts the occurrences in data packets.
- The second counter counts the occurrences in control packets.
- The third counter counts the occurrences in event packets.

Error counters management

The application uses the counters by means of Interface module feature of the CoaXPressErrorCounters Category.

Getting the current count value

1. Select a connector instance by setting the CxpHostConnectionSelector
2. Read the corresponding Interface module feature (e.g. CxpLinkLockLossCount)

Resetting a counter

1. Select a connector instance by setting the CxpHostConnectionSelector
2. Write to the corresponding Interface module feature (e.g. CxpLinkLockLossCountReset)

Related Interface module GenICam features

- CoaXPressErrorCounters parameters category
- CxpLinkLockLossCount and CxpLinkLockLossCountReset
- Cxp8b10bErrorCount and Cxp8b10bErrorCountReset
- CxpDuplicatedCharactersUncorrectedErrorCount and CxpDuplicatedCharactersUncorrectedErrorCountReset
- CxpDuplicatedCharactersCorrectedErrorCount and CxpDuplicatedCharactersCorrectedErrorCountReset
- CxpStreamDataPacketCrcErrorCount and CxpStreamDataPacketCrcErrorCountReset
- CxpControlPacketCrcErrorCount and CxpControlPacketCrcErrorCountReset
- CxpEventPacketCrcErrorCount and CxpEventPacketCrcErrorCountReset

3.11. CoaXPress Link Validation Tool

Introduction

Short Description

The *CoaXPress Link Validation Tool* (CXLVT) can be used to validate the operational parameters of a CoaXPress Link.

For a *quick test*, run the CXLVT until reaching a confidence level of 100% that the probability of single bit error (PER) is 10^{-10} or better 10^{-11}. This should just take a *few minutes*.

For an *extensive test*, run the CXLVT until reaching a confidence level of 100% that the PER is 10^{-12} or better 10^{-13}. This will take a *few hours*.

See also: http://en.wikipedia.org/wiki/Bit_error_rate for more information about the theory of bit error rate testing.

Host PC requirements

- The Host PC must be equipped with at least one EURESYS Coaxlink board.
- The Coaxlink driver must be installed on the Host PC.

Camera requirements

- The camera must be capable to generate a static image pattern.

Installation

The CXLVT is included in `gentl.exe`, a command-line tool that is delivered with the Coaxlink driver. No further installation is required.

gentl ber Command

The CXLVT is invoked with the command `ber` of `gentl.exe`.

```
$ gentl ber --help
GenTL Explorer

gentl ber [OPTIONS]
  Measure bit error rate confidence level (a.k.a. link validation tool)

Flags:
       --if=ID               Interface ID
       --dev=ID              Device ID
       --ds=ID               DataStream ID
       --buffers=INT         Buffer count (default: 4)
       --set=SETTINGS        GenApi settings, such as Module.Feature=INT
       --setup=FILE          Path to script to execute before starting stream
       --run=FILE            Path to script to execute concurrently with stream
       --remotexml=FILE      Use FILE as register description (default:
                             register description is read from remote device)
  -c   --create-only         Create a reference pattern and quit (requires
                             --output)
  -i   --input=FILE          Input reference pattern file (default:
                             automatically create a reference image before
                             measuring the bit error rate confidence level)
  -o   --output=FILE         Output reference pattern file (default: no output
                             file)
       --enable-dump=FILE    Enable dump of defective surfaces to files with
                             the given file path prefix
Common flags:
       --cti=LIBPATH         Path to GenTL producer library. Default: use
                             EURESYS_COAXLINK_GENTL64_CTI and
                             GENICAM_GENTL64_PATH environment variables to locate
                             the library.
  -j=N                       Limit the number of CPU cores to use to N
                             (default: 2)
  -h   --help                Display help message
  -V   --version             Print version information
       --numeric-version     Print just the version number
  -v   --verbose             Loud verbosity
  -q   --quiet               Quiet verbosity
```

gentl ber --help

```
> help
Ber commands:
  levels              show confidence levels
  levels -N           show confidence levels every N seconds
  results             show intermediate bit error rate results
  results -N          show intermediate bit error rate results every N
                      seconds
  report FILE         write current report to a file
  enable-dump FILE    enable dump of defective surfaces to files with the
                      given file path prefix
  disable-dump        disable dump of defective surfaces
General commands:
  quiet               set verbosity level to quiet
  normal              set verbosity level to normal
  loud                set verbosity level to loud
  help                display this help message
  exit                exit the CLI
>
```

gentl ber commands

Test procedure

To setup the *CoaxPress Link Validation Tool* proceed as follows:

1. With GenICam Browser:
   - Configure the camera as for normal operation and select a fixed test-pattern as video source
   - Configure the frame grabber as for normal operation
2. Open a command shell and execute `gentl ber` to start a Read-Eval-Print-Loop
3. Get intermediate results using the `results` command
   - Check if the number of acquired images counter increases regularly
   - Check the confidence levels
4. Run the test until the required confidence levels are reached. This may require several hours.

Operation

The CoaXPress Link Validation Tool (CXLVT) validates the operational parameters of a CoaXPress Link installation (bit rate, cable type, cable length) resulting in reliable, long-term performance.

CXLVT does this by estimating, with a known confidence level, the probability of single bit errors in a CoaXPress Link setup.

We define:

- *PER:* Probability of single bit error in a digital connection like a CoaXPress Link; this is an unknown quantity that we want to estimate
- *BER:* Bit Error Rate, actually measured by the CXLVT

It is generally accepted that a CoaXPress Link will operate reliably, if PER < 10^{-12}. This criterion is similar to the one used in other digital serial image transmission schemes. Of course, a better (lower) PER will provide even more assurance that the operation is reliable.

The CXLVT computes the confidence level (CL), or likelihood, that the PER is less than a set of values (10^{-10}, 10^{-11}, 10^{-12}, 10^{-13}, 10^{-14}), based on the measurement of the BER, during a time sufficiently long to accumulate the necessary evidence.

When started, the CXLVT displays these confidence levels, as evidence accumulates with the passing of time, as illustrated in the screenshots hereafter.

Entering the `levels` command, during the operation of the CXLVT, displays the confidence levels for the 5 PER values.

```
> levels
---------------------------------------------------------------------------------
Confidence level (rounded) that the probability of error is less          Elapsed
than:                                                                        Time
      1.0e-10      1.0e-11      1.0e-12      1.0e-13      1.0e-14      H:MM:SS
---------------------------------------------------------------------------------
       99.99%       64.97%        9.95%        1.04%        0.10%      0:08:56
>
```

Confidence levels reported 8 seconds after start

After 8 seconds, we have reached 99.99 % confidence level that the PER is less than 10^{-10}. The PER might very well be much better than that, but at this stage we have insufficient evidence to conclude that this is the case. The CXLVT must be continued.

By entering the `results` command, during the operation of the CXLVT, additional information can be displayed, after which the CXLVT continues its normal operation, for as long as necessary, to achieve the required confidence level for a predetermined PER.

```
> results
Intermediate results:
---------------------
Duration (hours:minutes:seconds):                              0:16:37
Duration (seconds):                                                997
Acquired images:                                                 11975
Bad images:                                                          0
Acquired bits:                                             1.986509e11
Bit errors:                                                0.000000e0
Average bit errors per bad image:                                    0
Bit rate (bits per second):                                1.992486e8
Bit error rate:                                            0.000000e0
Confidence level (rounded) that the probability of error
is less than 1.0e-10:                                          99.99%
             1.0e-11:                                          86.28%
             1.0e-12:                                          18.01%
             1.0e-13:                                           1.96%
             1.0e-14:                                           0.19%

>
```

Intermediate results reported after 16 minutes

From this screenshot, we can already conclude that the confidence level that the PER is less than 10^{-11} has risen from 64.97 % to 86.28 %, after 16 minutes.

The CXLVT should be continued until the confidence level that the PER is less than 10^{-12} (at most – a stronger test would be a PER less than 10^{-13}) has reached a satisfactory level (at least 95 %, and 99 % for a stronger result). This may require quite some time, because these outcomes require a significant amount of evidence.

```
Intermediate results:
---------------------
Duration (hours:minutes:seconds):                                     3:30:12
Duration (seconds):                                                     12612
Acquired images:                                                       149753
Bad images:                                                                 0
Acquired bits:                                                    2.484223e12
Bit errors:                                                       0.000000e0
Average bit errors per bad image:                                           0
Bit rate (bits per second):                                       1.969729e8
Bit error rate:                                                   0.000000e0
Confidence level (rounded) that the probability of error
is less than 1.0e-10:                                                 100.00%
             1.0e-11:                                                  99.99%
             1.0e-12:                                                  91.66%
             1.0e-13:                                                  21.99%
             1.0e-14:                                                   2.45%
```

Intermediate results reported after 3.5 hours

From this screenshot, we can conclude that the confidence level that the PER is less than 10^{-12} has risen from 18.01 % to 91.66 %, after 3.5 hours.

To generate a report, execute the `report` command.

```
> report ber-report
Report ber-report-20171107-093451.log successfully created
>
```

Generate a report command line

4. Image Data Path

4.1. FIFO Buffer

DRAM Memory Size per Product

Product	DRAM Memory Size
1629 Coaxlink Duo PCIe/104-EMB	512 MB
1630 Coaxlink Mono	512 MB
1631 Coaxlink Duo	1 GB
1632 Coaxlink Quad	1 GB
1633 Coaxlink Quad G3	1 GB
1633-LH Coaxlink Quad G3 LH	1 GB
1634 Coaxlink Duo PCIe/104-MIL	512 MB
1635 Coaxlink Quad G3 DF	1 GB
1637 Coaxlink Quad 3D-LLE	1 GB
3602 Coaxlink Octo	2 GB
3603 Coaxlink Quad CXP-12	2 GB
3620 Coaxlink Quad CXP-12 JPEG	2 GB

The DRAM memory is partitioned according to the installed firmware variants.

- All firmware variants allocate one partition named *FIFO Buffer* for each stream of each device.
- The firmware variants supporting *FFC* allocate one partition for the storage of gain and offset coefficients

Fifo Buffer Operation

The *Fifo Buffer* operates as a FIFO to decouple the CoaXPress data flow from the Pixel Processing and the PCI Express data flow.

It absorbs temporary dropouts of the PCI Express data flow ensuring a reliable CoaXPress data acquisition.

It enables burst-mode CoaXPress data acquisition at the highest data rates regardless the limits of the Pixel Processor and the PCI Express interface.

4.2. Acquisition Gate

The *Acquisition Gate* controls the image data extraction from the on-board FIFO buffer. It discards the image data that doesn't need to be acquired and fed to the"Pixel Data Processing" on the next page .

Area Scan Acquisition

The gate opens and closes at frame boundaries based on the application's calls of the `DSStartAcquisition` and `DSStopAcquisition` functions.

NOTE
The Camera and Illumination Controller indirectly controls the acquisition gating by issuing Camera Triggers using various schemes.

Line Scan Acquisition

The gate opens and closes at line boundaries according to the application `DSStartAcquisition` and `DSStopAcquisition` function calls and, according to the settings of the Image Acquisition Controller, to the Start-of-scan and the End-of-scan triggers.

See also: "Line Scan Acquisition" on page 236 for more information and configuration instructions.

4.3. Pixel Data Processing

The Image Pixel Data Processor performs the following successive operations on the image data stream:

CoaXPress bit stream slicing

This operation extracts individual pixel components data from the CoaXPress image data bit stream according to the bit depth – *input-bit-depth* – specified by the '*PixelF*' property of the CoaXPress Image Header.

All components have the same pixel bit depth. Possible values are 8-/10-/12-/14- and 16-bit.

The slicer delivers, for each image line, all the pixel components necessary to build a number of pixels specified by the '*Xsize*' property of the CoaXPress Image Header.

The slicer discards CoaXPress line-padding data.

Flat Field Correction

This operation applies a linear gain and offset transformation to each individual pixel components.

See also: "Flat Field Correction" on page 179 for more information and configuration instructions.

Lookup Table processing

This operation applies a lookup table transformation to each individual pixel components.

See also: "Lookup Table Processing" on page 193 for more information and configuration instructions.

Bayer CFA decoding

This operation transforms the raw Bayer CFA data stream issued by the camera into an RGB color data stream.

See also: "Bayer CFA Decoding" on page 206 for more information and configuration instructions.

Pixel component unpacking

This operation unpacks 10-bit, 12-bit, and 14-bit pixel components to 8-bit or 16-bit.

It can be disabled for monochrome and Bayer CFA pixel formats.

See also: "Pixel Component Unpacking" on page 69 for more information and configuration instructions.

Pixel component ordering

This operation modifies the component order of multi-components pixel data.

See also: "Pixel Component Re-Ordering" on page 71 for more information and configuration instructions.

Endianness conversion

This operation modifies the byte order of 16-bit pixel component data.

See also: "Endianness Conversion" on page 72 for more information.

Image line build-up

This operation builds concatenates the components data of all pixels of an image line:

- 8-bit pixel components are aligned to byte boundaries
- 16-bit pixel components (possibly expanded by unpacking or lookup table processing) are aligned to word (2-byte) boundaries, the 2 bytes are stored according to the little-endian convention.

Line padding

This operation appends padding bits or bytes to the image line data to reach the next alignment-boundary required by the hardware implementation.

The alignment boundary requirements are product-specific, for instance:

- 64-bit for **1630 Coaxlink Mono**, **1631 Coaxlink Duo**, and **1632 Coaxlink Quad**
- 128-bit for **1633 Coaxlink Quad G3**, **1633-LH Coaxlink Quad G3 LH** and **1635 Coaxlink Quad G3 DF**

Processing Performances

The pixel processor sustain the highest camera pixel rate. Unless specified otherwise, all the above operations are executed while transferring data to the GenTL with a negligible latency.

PCI Express Bandwidth Limitation

When acquiring pixels having a pixel bit depth larger than 8-bit, each pixel is expanded to 16-bit. In these cases, the PCI Express bandwidth limitation of the Host PC may negatively impact the achievable frame- or line-rate.

On-board Memory Bandwidth Limitation

For FFC use cases, the on-board memory bandwidth is not sufficient to sustain the full CoaXpress data rate.

4.4. Pixel Data Processing Configurations

This topic lists the applicable pixel data processing configurations for every class of camera pixel formats

Monochrome Pixel Formats - LUT disabled

Camera Pixel Format	FFC	Unpacking Mode	Output Pixel Format
Mono8	off | on	lsb | msb	Mono8
Mono10pmsb	off | on	lsb	Mono10
	off | on	msb	Mono16
	off	off	Mono10pmsb
Mono12pmsb	off | on	lsb	Mono12
	off | on	msb	Mono16
	off	off	Mono12pmsb
Mono14pmsb	off | on	lsb	Mono14
	off | on	msb	Mono16
	off	off	Mono14pmsb
Mono16	off | on	lsb | msb	Mono16

Monochrome Pixel Formats - LUT enabled

Camera Pixel Format	FFC	Lookup Table Configuration	Unpacking Mode	Output Pixel Format
Mono8	off | on	M_8x8	lsb | msb	Mono8
Mono10pmsb	off | on	M_10x8	lsb | msb	Mono8
		M_10x10	lsb	Mono10
			msb	Mono16
		M_10x16	lsb | msb	Mono16
Mono12pmsb	off | on	M_12x8	lsb | msb	Mono8
		M_12x12	lsb	Mono12
			msb	Mono16
		M_12x16	lsb | msb	Mono16

Bayer CFA Pixel Formats - CFA decoder disabled

Camera Pixel Format	FFC	Unpacking Mode	Output Pixel Format
Bayer**8	off | on	lsb | msb	Bayer**8
Bayer**10pmsb	off | on	lsb	Bayer**10
	off | on	msb	Bayer**16
	off	off	Bayer**10pmsb
Bayer**12pmsb	off | on	lsb	Bayer**12
	off | on	msb	Bayer**16
	off	off	Bayer**12pmsb
Bayer**14pmsb	off | on	lsb	Bayer**14
	off | on	msb	Bayer**16
	off	off	Bayer**14pmsb
Bayer**16	off | on	lsb | msb	Bayer**16

Bayer CFA Pixel Formats - CFA decoder enabled

Input Pixel Format	FFC	RedBlueSwap	Output Pixel Format
Bayer**8	off | on	off	RGB8
		on	BGR8
Bayer**10pmsb	off | on	off	RGB10
		on	BGR10
Bayer**12pmsb	off | on	off	RGB12
		on	BGR12
Bayer**14pmsb	off | on	off	RGB14
		on	BGR14
Bayer16	off | on	off	RGB16
		on	BGR16

RGB Pixel Formats

Input Pixel Format	FFC	Unpacking Mode	RedBlueSwap	Output Pixel Format
RGB8	off | on	lsb | msb	off	RGB8
			on	BGR8
RGB10pmsb	off | on	lsb	off	RGB10
			on	BGR10
		msb	off	RGB16
			on	BGR16
RGB12pmsb	off | on	lsb	off	RGB12
			on	BGR12
		msb	off	RGB16
			on	BGR16
RGB14pmsb	off | on	lsb	off	RGB14
			on	BGR14
		msb	off	RGB16
			on	BGR16
RGB16	off | on	lsb | msb	off	RGB16
			on	BGR16

RGBa Pixel Formats

Input Pixel Format	FFC	UnpackingMode	RedBlueSwap	Output Pixel Format
RGBa8	off | on	lsb | msb	off | on	RGBa8
RGBa10pmsb	off | on	lsb	off | on	RGBa10
		msb	off | on	RGBa16
RGBa12pmsb	off | on	lsb	off | on	RGBa12
		msb	off | on	RGBa16
RGBa14pmsb	off | on	lsb	off | on	RGBa14
		msb	off | on	RGBa16
RGBa16	off | on	lsb | msb	off | on	RGBa16

4.5. Pixel Component Unpacking

Introduction

The pixel data processor is capable of unpacking of 10-bit, 12-bit, and 14-bit pixel component data to 16-bit pixel data.

The unpacking operation is user-configurable through the UnpackingMode GenICam feature. Three options are available:

- Lsb: Unpacking to lsb (Default setting since release 4.3)
- Msb: Unpacking to msb
- Off: No unpacking

Unpacking to lsb

The significant bits of the pixel component data are aligned to the **least significant bit** of the data container. Padding '0' bits are put as necessary in the **most significant bits** to reach the next 8-bit boundary.

- 10-bit pixels: `0000 00<pp pppp pppp>`
- 12-bit pixels: `0000 <pppp pppp pppp>`
- 14-bit pixels: `00<pp pppp pppp pppp>`

NOTE
Unpacking to lsb doesn't modify the pixel component value.

Unpacking to msb

The significant bits of the pixel component data are aligned to the **most significant bit** of the data container. Padding '0' bits are put as necessary in the **least significant bits** to reach the next 8-bit boundary.

- 10-bit pixels: `<pppp pppp pp>00 0000`
- 12-bit pixels: `<pppp pppp pppp> 0000`
- 14-bit pixels: `<pppp pppp pppp pp>00`

TIP
Unpacking 10-bit, 12-bit, and 14-bit pixel components to msb multiplies the pixel component value by 64, 16, and 4 respectively.

NOTE Unpacking 8-bit and 16-bit pixel components is a neutral operation:

- The size of the data container is unchanged: One byte for 8-bit pixel components; two bytes for 16-bit pixel components.
- The data bits are not modified.

WARNING
Unpacking 10-bit, 12-bit, and 14-bit pixel components increases the amount of data by 160%, 133%, and 114% respectively!

No unpacking

The packed image data transmitted by the camera through the CoaXPress Link is delivered as is to the output buffer.

WARNING
This option is only available with area-scan firmware variants for cameras delivering monochrome and Bayer CFA pixels!

4.6. Pixel Component Re-Ordering

The Coaxlink image data stream pixel processor can be configured to swap the first and the third component data of 3-component pixels.

The swapping is controlled through the RedBlueSwap boolean GenICam feature:

- When set to False (default settings), the original component order is preserved
- When set to True, the first and the third components are swapped.

The function is available for image acquisition from:

- RGB color cameras delivering 3-component pixel data,
- BAYER CFA color cameras providing that the BAYER CFA decoding is enabled.

4.7. Endianness Conversion

The Coaxlink image data stream pixel processor delivers 16-bit pixel components using the little-endian convention.

The conversion is not performed when UnpackingMode is set to Off.

Little-endian Convention

The least-significant byte of a multiple byte data is stored at the lowest address location.

For instance, 16-bit data are stored into two consecutive byte locations as follows:

Memory Byte Location	Memory Byte Content
N	Data[7:0]
N+1	Data[15:8]

4.8. Pixel Ordering

The Coaxlink image data stream pixel processor preserves the pixel order of the CoaXPress data stream.

The pixels data of an image frame are stored in successive address locations starting with the first pixel of the first line at the lowest address.

The successive lines of an image frame are concatenated in the image buffer.

4.9. Image Data Transfer

Buffer Filling

A DMA engine transfers the processed image data over the PCI Express bus to the allocated GenTL buffers according to rules that are different for line-scan and area-scan image acquisition.

GenTL Buffer Filling Rules – Area-scan cameras

In area-scan imaging, GenTL buffers are filled according to the following rules:

- The first acquired line data of a frame is, by default, stored at the beginning of a new buffer. When vertical image flipping is enabled by setting StripeArrangement to Geometry_1X_1YE, the first acquired line data of a frame is stored at the location of the last full line of a new buffer.
- When image transfer to host memory is done, the buffer, possibly partially filled, is made available to the application for processing.
- When the buffer is too small to contain a complete frame, the remaining data is discarded.

GenTL Buffer Filling Rules – Line-scan cameras

In line-scan imaging, GenTL buffers are filled according to the following rules:

- The first acquired line data of a scan is, by default, stored at the beginning of a new buffer. When vertical image flipping is enabled by setting StripeArrangement to Geometry_1X_1YE, the first acquired line data of a scan is stored at the location of the last full line of a new buffer.
- A buffer contains an integer number of image lines data.
- When the remaining space of a buffer is not sufficient to store an image line data, the acquisition continues into a new buffer and the filled buffer is made available to the application for processing.
- When the last line data of a scan is acquired, the last buffer, possibly partially filled, is made available to the application for processing.

Image data padding

The DMA engine provides the capability to organize the data differently in the buffer by adding line padding or stripe padding.

> **NOTE**
> Prior to Coaxlink driver 6.2, the DMA engine was transferring the whole image data as a single 1D entity regardless the 2D structure: the lines of processed image data are concatenated into the destination buffer.

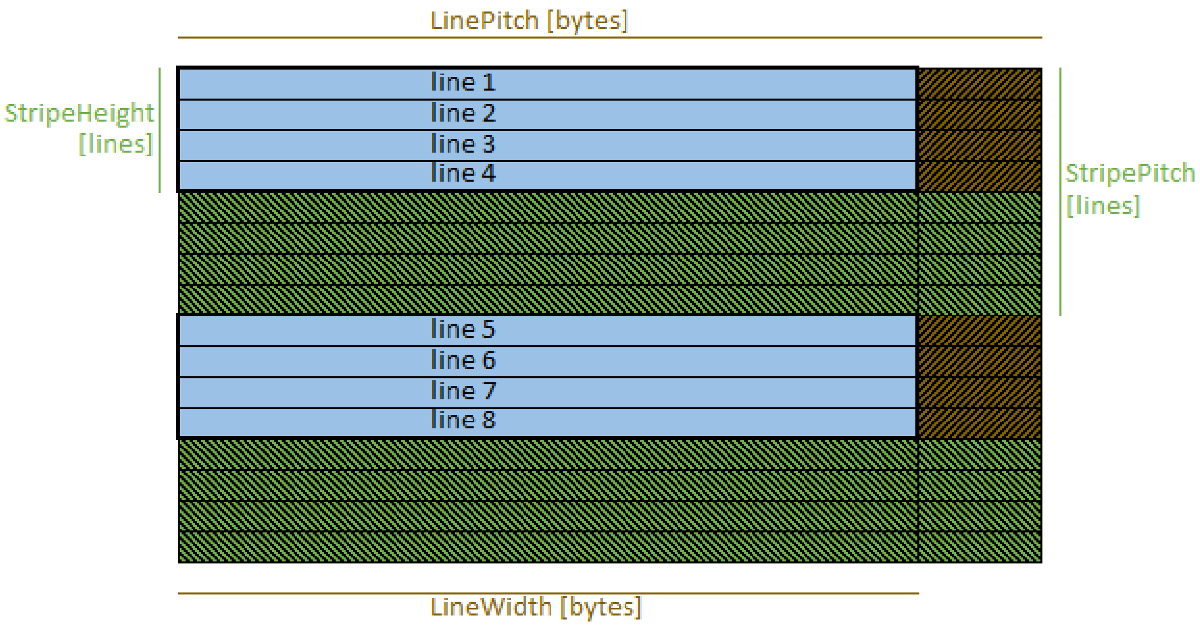


Image buffer padding model

Line Padding

The LineWidth and LinePitch features control the line padding.

When LinePitch > LineWidth, the line padding is enabled: the DMA engine leaves LinePitch - LineWidth bytes of padding at the end of each image line.

LinePitch can be set to 0 to disable padding after lines.

Stripe padding

Stripes are groups of adjacent lines. A stripe of height 1 is a line.

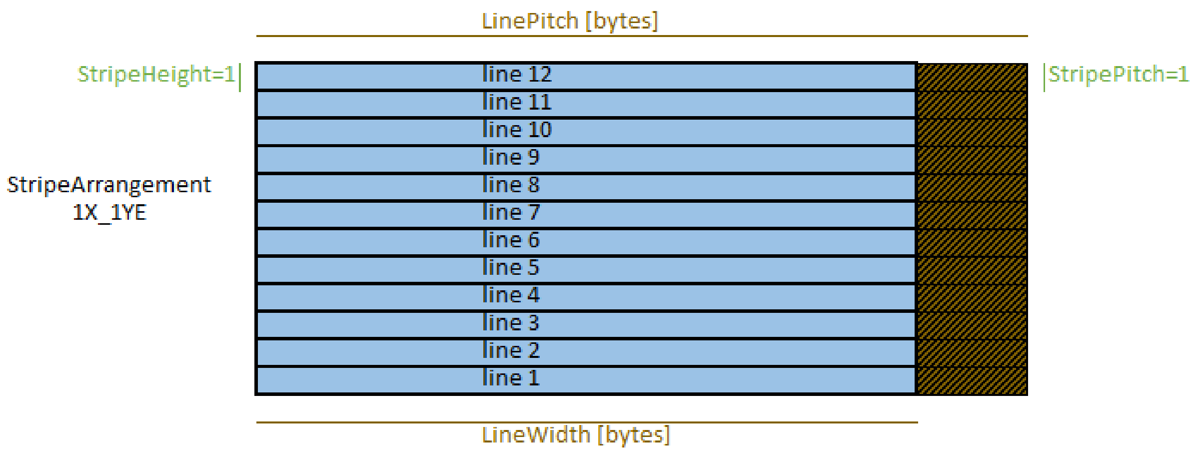
The StripeHeight and StripePitch features control the stripe padding.

When StripePitch > StripeHeight, the stripe padding is enabled: the DMA engine leaves StripePitch - StripeHeight lines of padding at the end of each stripe.

StripePitch can be set to 0 to disable padding after lines.

Vertical Image Flip

The DMA engine provides the capability to flip the image vertically.



Flipped image data

The vertical image flip is controlled by the StripeArrangement feature of the data Stream module.

By default, StripeArrangement is set to 1X_1Y: the vertical image flip is disabled.

When StripeArrangement is set to 1X_1YE, the driver determines the position of the first image line in the buffer by using this formula:

BufferBase + (BufferSize + LinePitch - LineWidth) / LinePitch * LinePitch - LinePitch.

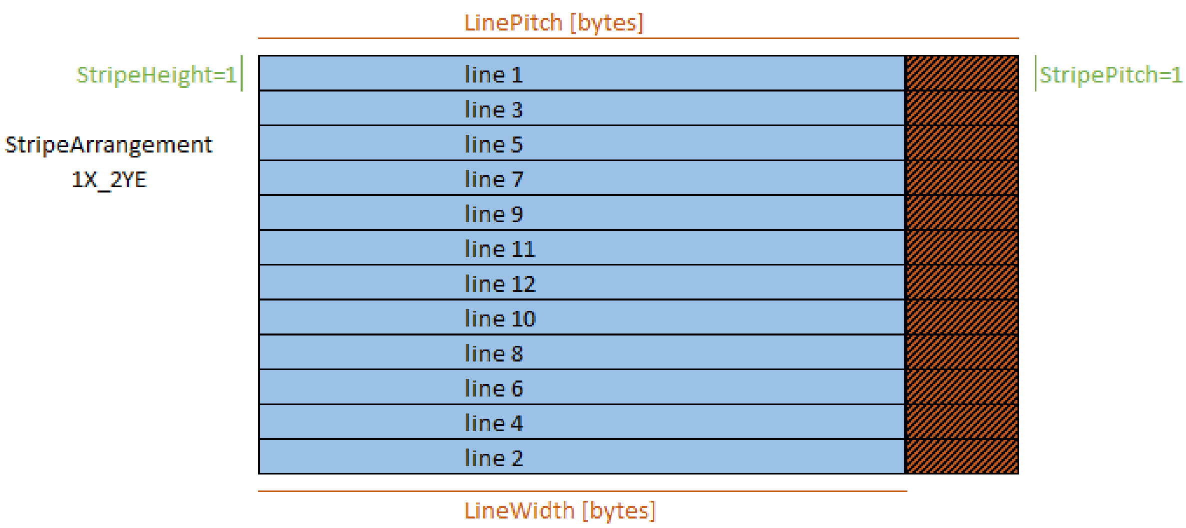
As a result:

- if the buffer is too small, it is the bottom part of the image (as given by the camera) that will be lost;
- lines will start at BufferBase + n * LinePitch;
- only complete lines are transferred;
- if the buffer size is not a multiple of LinePitch bytes, some bytes at the end of the buffer will be left unchanged.

> **NOTE**
> When evaluating the above formula, if LinePitch is equal to 0, LineWidth will be used instead. Similarly, if StripeHeight is 0, 1 will be used instead.

Unscrambling 1X_2YE images

The DMA engine provides the capability to unscramble 1X_2YE images.



When StripeArrangement is set to Geometry_1X_2YE, the driver determines the destination of the second line output by the camera (i.e., the position of last image line in the buffer) by using this formula:

BufferBase + (BufferSize + LinePitch - LineWidth) / LinePitch * LinePitch - LinePitch

NOTE
this is the address of the last line in the buffer large enough to receive one complete line.

As a result:

- If the buffer is too small, the last lines output by the camera (i.e., the middle part of the image) will be lost; the application is responsible for avoiding this,
- Lines will start at

```
BufferBase + n * LinePitch,
```

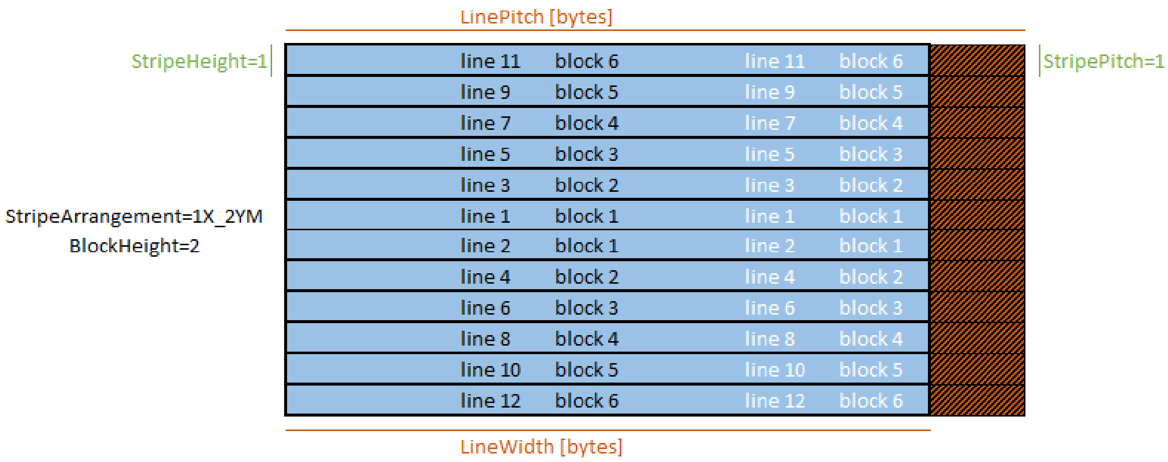
- Only complete lines are transferred. (When evaluating the above formula, if LinePitch is equal to 0, LineWidth will be used instead. Similarly, if StripeHeight is 0, 1 will be used instead.)
- StripeHeight and StripePitch cannot be set to values greater than 1.

Unscrambling 1X_2YM images

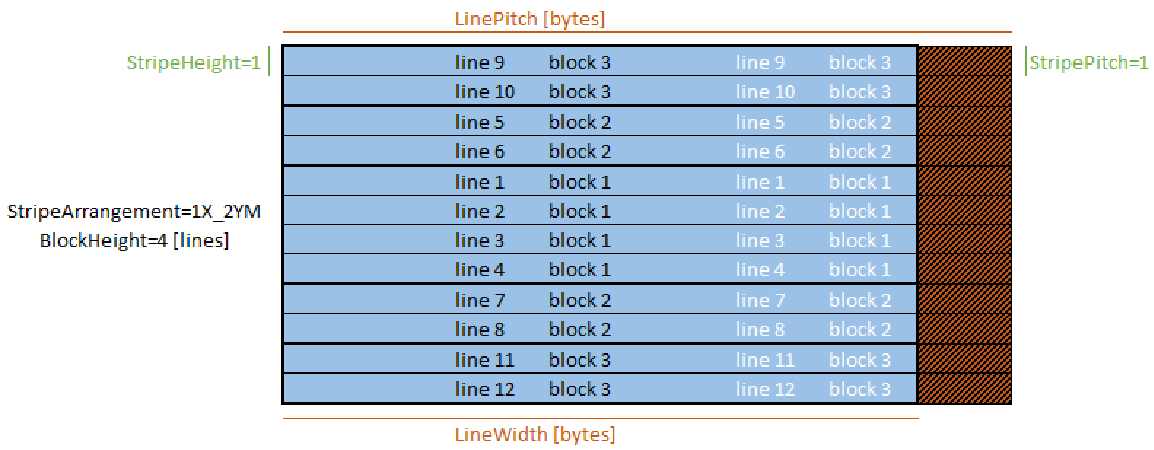
The DMA engine provides the capability to unscramble 1X_2YM images.

> **NOTE**
> In the following figures, line numbers and block numbers sent by the *device* are in *white*, line numbers and block numbers received by the *host* are in *black*.

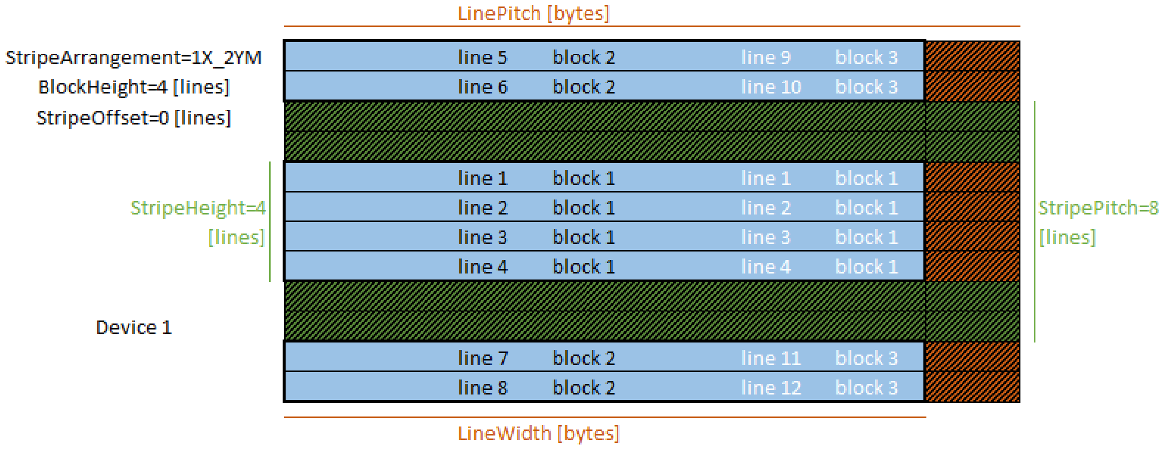
1X_2YM camera delivering lines by blocks of 2

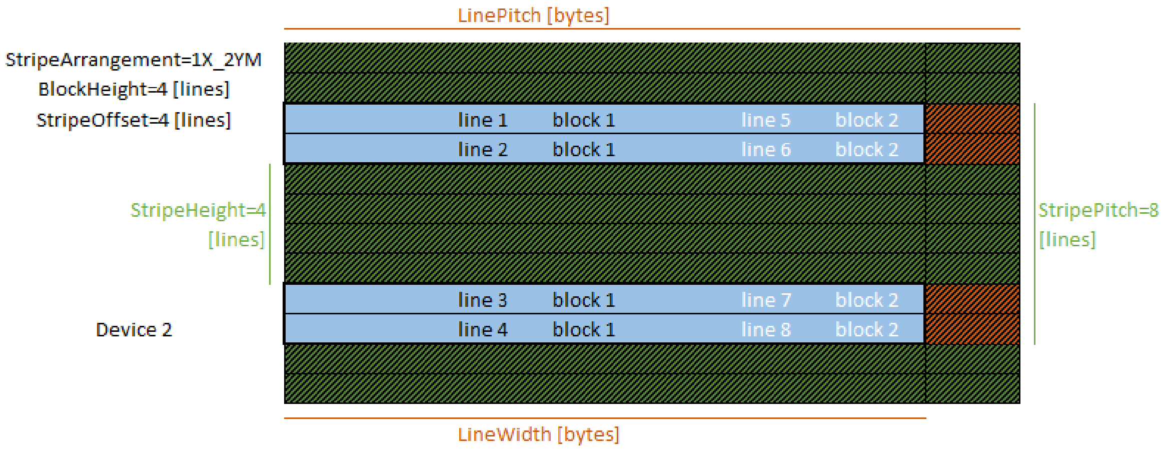


1X_2YM camera delivering lines by blocks of 4



1X_2YM camera delivering lines by blocks of 4 to two hosts





Statistics

The stream statistics tool monitors the image data stream at the card output and provides the application with averaged frame-, line- and data-rate.

Stream Statistics Sampling Methods

The StatisticsSamplingSelector determines the *averaging interval*. It can be any of the following:

- LastSecond or LastTenSeconds: The last completed *time* slot of 1 or 10 seconds.
- Last2Buffers, Last10Buffers, Last100Buffers, Last1000Buffers: The last 2, 10, 100, or 1000 *acquired buffers*
- LastAcquisition: The last acquisition activity period. Namely since the last DSStartAcquisition() function call until now, if the acquisition is still active otherwise until the last DSStopAcquisition() function call.
- LastAcquisition: Time interval between StatisticsStartSampling and StatisticsStopSampling commands.

The default sampling method is LastSecond.

Statistical Data

The statistical data is effectively computed when getting any of the following feature:

- StatisticsFrameRate reports the averaged frame rate expressed in in frames/second (area-scan).
- StatisticsLineRate reports the average line rate expressed in lines/second (line-scan).
- StatisticsDataRate reports the average data rate expressed in megabytes/second

For every GenTL buffer filled during the averaging interval, the tool counts:

- The number of filled GenTL buffers and the corresponding number of frames (area-scan) or lines (line-scan)
- The number of transferred bytes of image data.

The related GenICam features rare gathered into the Stream Statistics Category of the GenTL Data Stream Module.

4.10. Data Transfer Rate Test Program

Introduction

The *Data Transfer Rate Test Program* (DTR) can be used to measure the effective PCI Express data transfer rate in real conditions.

Host PC requirements

- The Host PC must be equipped with at least one EURESYS Coaxlink board.
- The Coaxlink driver version 12.4 or higher must be installed on the Host PC.

Camera requirements

- The camera must be configured to deliver continuously image data.

Installation

The DTR is included in `gentl.exe`, a command-line tool that is delivered with the Coaxlink driver. No further installation is required.

Measurement principle

The DTR measures the data transfer rate by completely filling the internal frame store and only then transferring images to the host computer:

1. All buffers are unqueued (the data stream cannot use them)
2. The data stream and remote device are started
3. When the frame store is full, the remote device is stopped
4. Current timestamp is retrieved (t0)
5. All buffers are queued to the data stream and transfers start
6. Buffers are popped from the data stream
7. When the frame store is empty and all buffers have been retrieved, the data stream is stopped

The DTR program computes the `data transfer rate` as follows:

```
- byte count = sum of each buffer's BUFFER_INFO_SIZE_FILLED
- t1 = last buffer's BUFFER_INFO_CUSTOM_EVENT_TIMESTAMP
- duration = t1 - t0
- data transfer rate = byte count / duration
```

gentl --help

```
GenTL Explorer

gentl [COMMAND] ... [OPTIONS]

Commands:
       info    Show detailed information about the transport layer system
       report  Generate a GenTL report archive (for Euresys tech support)
       xml     Download GenApi files (XML register descriptions)
       play    Open a data stream and acquire images (no display)
       view    Open a data stream and display images
       grab    Grab N images
       genapi  Enter the GenApi command-line interface or perform a GenApi operation
       read    Read data from a GenTL port
       write   Write data to a GenTL port
       event   Wait for events and display information about them
       script  Execute script
       run     Run an action
       dtr     Measure PCIe data transfer rate
       ber     Measure bit error rate confidence level (a.k.a. link validation tool)

Common flags:
               --cti=LIBPATH            Path to GenTL producer library.
                                        Default: use EURESYS_COAXLINK_GENTL64_CTI and
                                        GENICAM_GENTL64_PATH environment variables to
                                        locate the library.
       -j=N                             Limit the number of CPU cores to use to N (default: 2)
       -h      --help                   Display help message
       -V      --version                Print version information
               --numeric-version        Print just the version number
       -v      --verbose                Loud verbosity
       -q      --quiet                  Quiet verbosity
```

gentl dtr --help

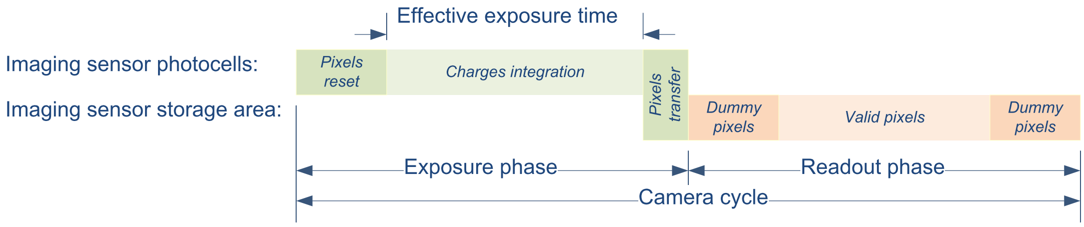
```
GenTL Explorer

gentl dtr [OPTIONS]

Flags:
       --if=ID                         Interface ID
       --dev=ID                        Device ID
       --ds=ID                         Data stream ID
       --device-access=ACCESS          Access flags used to open the device (GenTL standard access
flags:
                                       DEVICE_ACCESS_READONLY, DEVICE_ACCESS_CONTROL, DEVICE_
ACCESS_EXCLUSIVE;
                                       Coaxlink custom access flags:
                                       DEVICE_ACCESS_CUSTOM_READONLY_DEVICE_READONLY_STREAM)
(default:
                                       DEVICE_ACCESS_CONTROL)
       --ro                            Open the device as read-only (shorthand for
                                       --device-access=DEVICE_ACCESS_READONLY)
       --buffers=INT                   Buffer count (default: 4)
       --buffersize=INT                Buffer size
       --width=WIDTH                   Buffer width
       --height=HEIGHT                 Buffer height
       --pixelformat=ITEM              PFNC Pixel format
       --bayer=BAYERDECODINGMETHOD     Bayer method (Legacy, Advanced) (default: Advanced)
       --set=SETTINGS                  GenApi settings, such as Module.Feature=INT
       --setup=FILE                    Path to script to execute before starting stream
       --run=FILE                      Path to script to execute concurrently
       --timeout=INT                   Acquisition timeout, in milliseconds (default: infinite)
       --zero                          Zero memory when queuing buffers (default: memory is only
zeroed when buffers
                                       are allocated)
       --remotexml=FILE                Use FILE as register description (default: register
description is read from
                                       remote device)
  -n   --repeat[=N]                    Measure data transfer rate N times (default: 1)
Common flags:
       --cti=LIBPATH                   Path to GenTL producer library. Default: use EURESYS_
COAXLINK_GENTL64_CTI and
                                       GENICAM_GENTL64_PATH environment variables to locate the
library.
  -j=N                                 Limit the number of CPU cores to use to N (default: 2)
  -h   --help                          Display help message
  -V   --version                       Print version information
       --numeric-version               Print just the version number
  -v   --verbose                       Loud verbosity
  -q   --quiet                         Quiet verbosity
```

TIP For a better measurement accuracy, use the `gentl ber -n` option to execute multiple measurements repeatedly. The DTR program will average the results.

5. Camera Control Principles

5.1. Camera Cycle

A camera cycle is composed of two consecutive phases: the exposure phase and the readout phase.



Camera Cycle

Exposure phase

The exposure phase is the period of time during which the photocells of the imaging sensor integrate electric charges induced by the incoming photons.

For cameras having an electronic shutter, the exposure phase begins with a pixel reset action that clears all the sensor photocells. For permanent exposure cameras, i.e. cameras having no (or not using) the electronic shutter, the exposure phase begins immediately after the completion of the previous exposure phase.

For all types of cameras, the exposure phase terminates with a “pixel transfer” action. The accumulated charges in the photocell are transferred to the storage area for further readout. This action clears the photocells and new charge integration begins immediately.

Cameras having an electronic shutter have the capability to reset the pixels asynchronously and initiate a new exposure on request. These cameras are named asynchronous reset cameras.

Having the capability of controlling the time of the start of exposure (pixel reset) and the time of the end of exposure (pixel transfer) gives full control on:

- The timing of each image capture
- The sensitivity of the imaging sensor by selecting the exposure time

Readout phase

The readout phase is the period of time during which the total amount of electrical charges accumulated by each pixel is measured and delivered to the imaging sensor output.

The readout phase is not controlled by the frame grabber:

- It is automatically initiated after each pixel transfer.
- Its duration is fixed; it is determined by the amount of pixel data to be transferred and by the readout structure of the sensor (one or more taps, tap output data rate).

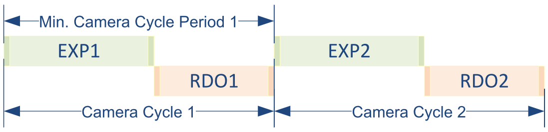
Some sensors provide the capability to select one or more region of interest (ROI) speeding up the readout since less data needs to be transferred.

5.2. Camera Cycle Concatenation Rules

This topic explains the rules that MUST be observed by the frame grabber to avoid *Camera Trigger* overrun when requesting successive camera cycles to an asynchronous reset camera.

Rule for cameras not allowing overlapping

- The next camera cycle may NOT begin before the completion of the readout phase.



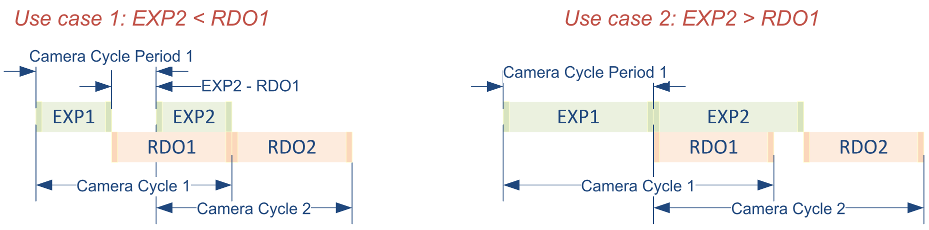
Shortest possible cycle period achievable by cameras NOT allowing the cycle overlapping

$$Min\ Cycle\ Period_n = EXP_n + RDO_n$$

> **NOTE**
> Only a minority of industrial cameras are NOT allowing the cycle overlapping!

Rules for cameras allowing overlapping

1. The exposure phases of two consecutive camera cycles may NEVER overlap.
2. The readout phases of two consecutive camera cycles may NEVER overlap



Shortest possible cycle period achievable by cameras allowing cycle overlapping

In the first case, the duration of the exposure phase of the 2nd cycle is shorter than the duration of the readout phase of the first cycle. The next camera cycle may start (EXP_{n+1} - RDO_n) period of time after the completion of the exposure phase. The minimum cycle period is

$$Min\ Cycle\ Period_n = EXP_n + RDO_n - EXP_{n+1}$$

In the second case, the duration of the exposure phase of the 2nd cycle is longer than the duration of the readout phase of the first cycle. The next camera cycle may start immediately after the completion of the exposure phase. The minimum cycle period is:

$$Min\ Cycle\ Period_n = EXP_n$$

NOTE
The majority of asynchronous reset cameras used in the machine vision industry supports the overlapping of the camera cycles!

5.3. Camera Control Methods

Coaxlink frame grabbers provide four camera control methods named NC, RC, RG and EXTERNAL.

NC Camera Control Method

The NC camera control method targets cameras that are NOT controlled by the frame grabber. This includes

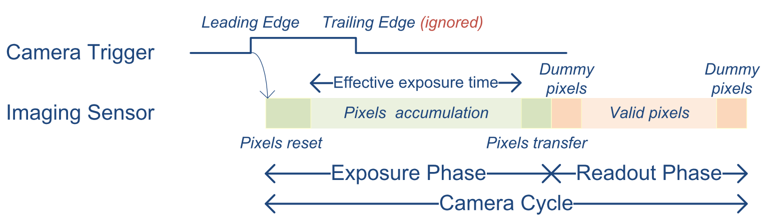
- Free-run cameras not using any external trigger signal
- Asynchronous-reset cameras using an external trigger signal not delivered by the frame grabber.

> **NOTE**
> The Camera and Illumination Controller (CIC) of Coaxlink is not used. There are no real-time control of the camera by the frame grabber. There are no Camera Trigger signal produced by Coaxlink. There are no real-time control of the illumination. There are no Strobe signal produced by Coaxlink.

RC Camera Control Method

The RC camera control method targets asynchronous reset cameras where only the camera cycle rate is controlled by the frame grabber. The exposure duration is controlled by the camera.

The real-time control is performed through a single upstream signal named "*Camera Trigger*" issued by the Camera and Illumination Controller (CIC) of Coaxlink.



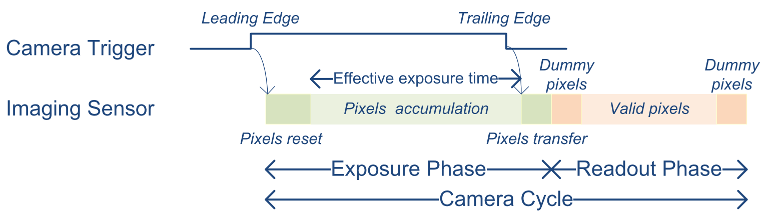
Grabber controlled camera cycle using the RC camera control method

The CIC produces one single *Camera Trigger* pulse every camera cycle. The *Camera Trigger* leading edge triggers a new camera cycle and initiates a new exposure period. The *Camera Trigger* trailing edge is ignored by the camera.

RG Camera Control Method

The RG camera control method targets asynchronous reset cameras where both the camera cycle rate and the exposure duration are controlled by the frame grabber.

The real-time control is performed through a single upstream signal named *Camera Trigger* issued by the Camera and Illumination Controller (CIC) of Coaxlink.



Grabber controlled camera cycle using the RG camera control method

The CIC produces one single Camera Trigger pulse every camera cycle. The *Camera Trigger* leading edge triggers a new camera cycle and initiates a new exposure period. The *Camera Trigger* trailing edge terminates the exposure period and triggers the readout.

EXTERNAL Camera Control Method

The EXTERNAL camera control method targets cameras that are NOT controlled by the frame grabber. This includes

- Free-run cameras not using any external trigger signal
- Asynchronous-reset cameras using an external trigger signal not delivered by the frame grabber.

The EXTERNAL camera control method targets asynchronous reset cameras that are controlled by a hardware signal applied by an external controller to any GPIO input port of the grabber.

NOTE
The Camera and Illumination Controller (CIC) of Coaxlink is not used. There are no real-time control of the camera by the frame grabber. There are no Camera Trigger signal produced by Coaxlink. There are no real-time control of the illumination. There are no Strobe signal produced by Coaxlink.

WARNING
The external controller is entirely responsible for the camera cycle timings!

6. Illumination Control Principles

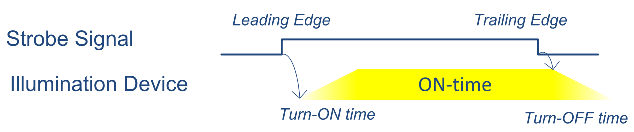
6.1. Illumination Devices

Two classes of illumination devices can be controlled by the illumination controller:

- Intermittent illumination devices
- Strobed illumination devices

Intermittent Illumination Devices

This illumination device class includes switched light sources where the turn-on and the turn-off time are controlled by the leading and the falling edges of the strobe signal.



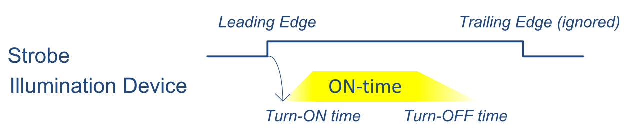
Timing diagram for intermittent illumination devices

The width of the strobe pulse determines the ON time duration of the light source

> **NOTE**
> The turn-on time and the turn-off time need to be considered when configuring the illumination controller!

Strobed Illumination Devices

This illumination device class includes switched light sources where only the turn-on time is controlled by the leading edge of the strobe signal.



Timing diagram for strobed illumination devices

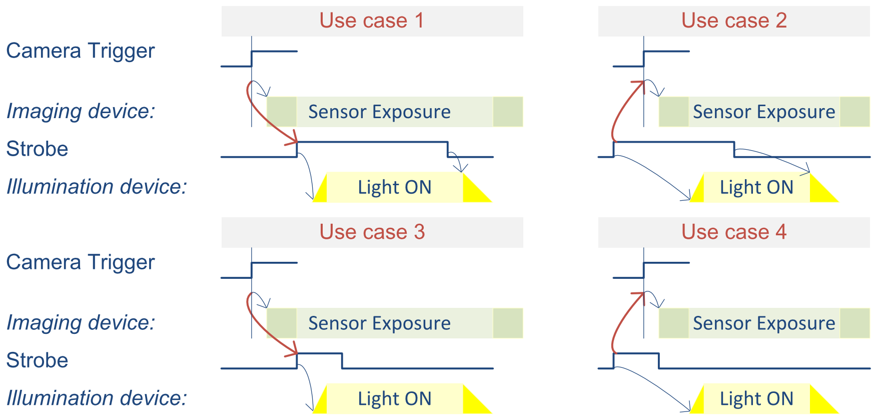
The on-time duration is either uncontrolled or controlled by the illumination device itself.

NOTE
The turn-on time and the ON time duration need to be considered when configuring the illumination controller.

6.2. Aligning Camera and Illumination Cycles

Obviously, the ON time of the light source must coincide with the exposure phase of the imaging sensor.

Therefore, the time relationship between the strobe signal(s) and the Camera Trigger signal must be adequately controlled.



4 typical use cases of Camera Trigger vs. Strobe alignment

Intermittent Light Sources (Use cases 1 & 2)

The duration of the strobe pulse must be adequately controlled in order to provide the right amount of light and get a correctly exposed image.

The sensor exposure should be adequately timed in order to terminate the sensor exposure after the light has turned off.

Strobed Light Sources (Use cases 3 & 4)

The sensor exposure should be adequately timed in order to terminate the sensor exposure after the light has turned off.

Late Strobe (Use cases 1 & 3)

The leading edge (beginning) of the Strobe signal is delayed a little to ensure that the light is not turned on too early while the imaging device is resetting its pixels.

Early Strobe (Use cases 2 & 4)

The leading edge of the Camera Trigger signal is delayed a little to ensure that the sensor exposure time is kept as short as possible and closely matches the on time.

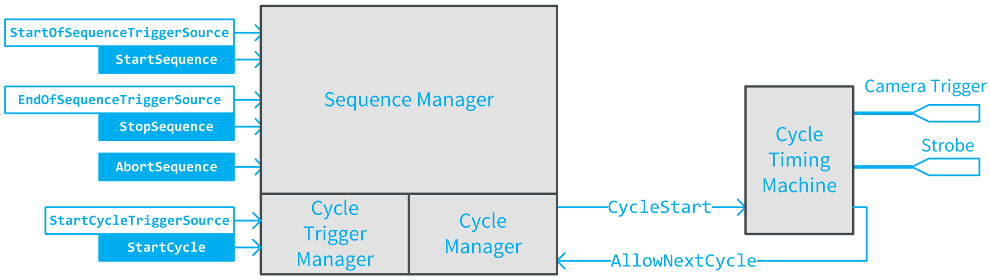
7. Camera and Illumination Controller

7.1. CIC Block Diagram

The Camera and Illumination Controller (abbreviated as CIC) controls one camera and its associated illumination.

The camera is controlled with the Camera Trigger signal and the illumination device is controlled with the Strobe signal.

See also: "CIC Output Signals Routing" on page 105



Camera and Illumination Controller block diagram

The CIC is composed of 4 main interconnected blocks:

- The *Cycle Timing Machine* is responsible for the generation of accurately timed events and signals structuring one camera and illumination controller cycle (CIC Cycle), namely: Camera Trigger and Strobe.

See also: "Cycle Timing Machine" on the next page

- The *Cycle Manager* is responsible for the generation of the CycleStart event. It prevents initiating a new cycle while the *start cycle conditions* are not all satisfied and while the cycle timing machine does not allow a new cycle to begin.

See also: "Cycle Manager" on page 100

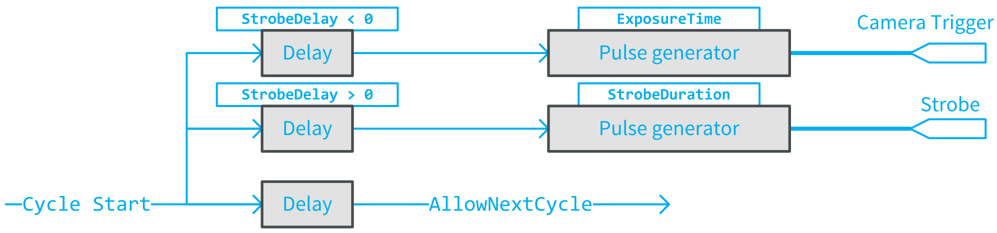
- The *Cycle Trigger Manager* is responsible, in collaboration with the *Cycle Manager* and the *Sequence Manager*, to elaborate the effective CycleStart event that initiates one cycle of the *Cycle Timing Machine*.

See also: "Cycle Trigger Manager" on page 101

- The *Sequence Manager* manages sequences of CIC cycles according to user-defined start sequence and stop sequence conditions.

See also: "Sequence Manager" on page 103

7.2. Cycle Timing Machine



CIC Cycle Timing Machine block diagram

The CIC timing machine is responsible for the generation of accurately timed events and signals structuring one camera and illumination controller cycle (CIC Cycle).

At every occurrence of a Cycle Start event, the timing machine generates:

- One single pulse on the Camera Trigger signal.
- One single pulse on the Strobe signal.
- One AllowNextCycle event.

Intra-cycle Timing

Three GenICam features of the Device module are used to configure the timing of the output signals within a cycle:

- ExposureTime defines the duration of the Camera Trigger pulse.
- StobeDuration defines the duration of the Strobe pulse.
- StrobeDelay defines the time offset from the leading edge of Camera Trigger up to the leading edge of Strobe.

See also: "Cycle Timing Diagrams" on page 107 for more explanations and timing diagrams

Cycle-to-Cycle Timing

The AllowNextCycle event is used by the Cycle Manager to determine when the next Cycle may start.

The position of the AllowNextCycle event is not directly set by the user. Instead, it is evaluated by the Coaxlink driver according to the following user settings:

- The ExposureReadoutOverlap feature of the Camera Model category defines if the camera supports or not the exposure/readout overlapping. If overlapping is allowed, the AllowNextCycle event is issued earlier and faster cycle rates are obtained.
- The ExposeRecovery feature of the Camera Model category defines the minimum time gap required by the camera between two exposures. This feature is relevant when ExposureReadoutOverlap = TRUE and the duration of the exposure phase becomes larger than the duration of the readout phase.
- The CycleMinimumPeriod of the Cycle Control category defines the minimum cycle period. This value may not be smaller than the time required by the camera to perform the image readout!

NOTE
Some cameras have a data store in the image data path. This enables capturing bursts of images at a higher cycle-to-cycle rate than the camera-to-frame grabber data link can sustain. In that case, CycleMinimumPeriod declares the smallest cycle-to-cycle time that the image sensor can achieve!

See also: "Consecutive Overlapping CIC Cycles" on page 109 for more explanations and timing diagrams.

7.3. Cycle Manager

The Cycle Manager is responsible for the generation of the Cycle Start event.

It prevents initiating a new cycle while the *cycle start conditions* listed hereafter are not satisfied:

Sequence Active Condition

The Sequence Manager must be in the ACTIVE state: the sequence has started and the sequence stop condition has not yet been reached.

This condition always applies.

Next Cycle Allowed Condition

The Cycle Timing Machine has already issued the Allow Next Cycle event of the previous cycle.

This condition always applies.

Free Memory Condition

There is enough free memory on board to acquire the image data of the next cycle.

This condition always applies.

Cycle Trigger Event Condition

A new Cycle Trigger event is required to initiate a new cycle.

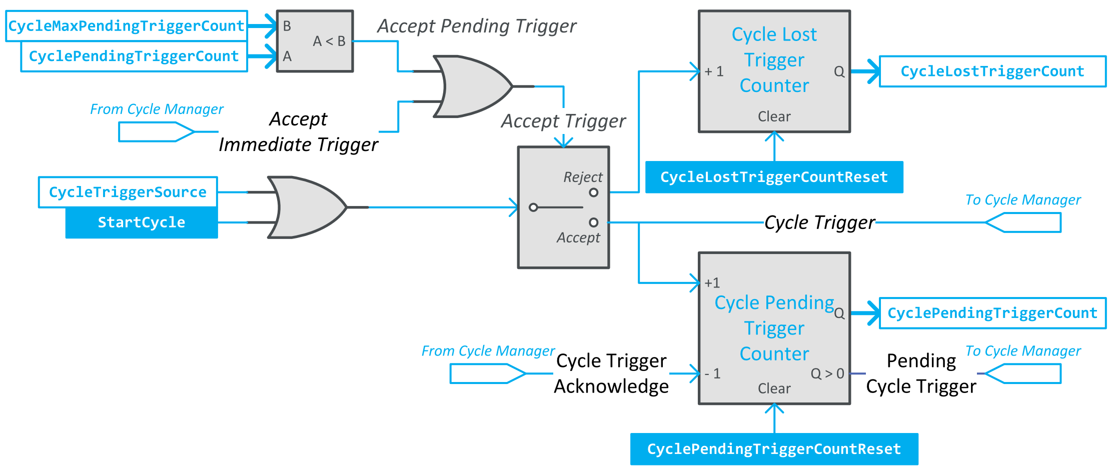
This condition applies only when Cycle TriggerSource ≠ Immediate AND CycleMaxPendingTriggerCount = 0.

Pending Trigger Condition

There is at least one pending trigger (possibly a new one) waiting for service.

This condition applies only when Cycle TriggerSource ≠ Immediate AND CycleMaxPendingTriggerCount > 0.

7.4. Cycle Trigger Manager



CIC Cycle Trigger Manager block diagram

The Cycle Trigger Manager is responsible, in collaboration with the Cycle Manager, to elaborate the effective *Cycle Trigger* event that initiates one cycle of the CIC timing machine.

Cycle Trigger Sources

The source of Cycle Trigger is defined by CycleTriggerSource.

When set to Immediate, the Cycle Trigger Manager is self-triggered. It generates a Cycle Trigger immediately after the start of the sequence and then repeatedly every CycleMinimumPeriod period.

When set to StartCycle or to *<any I/O toolbox event source>* the Cycle Trigger Manager doesn't start immediately after the start of the sequence, instead it waits for the execution of a StartCycle command or the occurrence of an event on the selected I/O toolbox event source.

A wide set of Cycle Trigger event sources is available. It includes all the I/O toolbox events, namely: LIN, QDC, MDV, DIV, DEL, EIN and User Events.

Cycle Trigger Latch Mechanism

The Cycle Manager is fitted with a trigger latch mechanism capable of latching cycle triggers that cannot be served immediately. Such triggers are named ''pending triggers" since their execution is simply postponed until the corresponding CIC cycle is initiated.

The maximum number of pending triggers that can be recorded is defined by CycleMaxPendingTriggerCount. When CycleMaxPendingTriggerCount = 0, the trigger latching mechanism is disabled. This is the default value. To enable the trigger latching mechanism, set CycleMaxPendingTriggerCount to any integer value in range 1 to 7.

The number of pending triggers is reported by CyclePendingTriggerCount.

Cycle Trigger Events Sorting

When CycleTriggerSource is set to StartCycle or to *<any I/O toolbox event source>*, every Cycle Trigger event is evaluated against the *trigger acceptance criteria* and sorted according to the result.

The rejected Cycle Trigger events increment the *Cycle Lost Trigger Counter*.

The accepted Cycle Trigger events increment the *Cycle Pending Trigger Counter* if the pending trigger cannot be served immediately.

Trigger Acceptance Criteria

Cycle Trigger events are *accepted and executed immediately* when both conditions are satisfied:

- Cycle Sequence is active
- Cycle Manager is currently waiting for an immediate trigger event to start a new cycle (Accept Immediate Trigger)

Cycle Trigger events are *accepted and executed later* when following conditions are satisfied:

- Cycle Sequence is active.
- The number of pending triggers, CycleMaxPendingTriggerCount, is less than CycleMaxPendingTriggerCount.
- The Cycle Manager is not (yet) ready to initiate new cycle.

7.5. Sequence Manager

The *Sequence Manager* is the top-level manager of the CIC: It controls the Cycle Trigger Manager and the Cycle Manager.

If defines sequences of identical CIC cycles according to user-defined start sequence and stop sequence conditions.

Starting a Sequence

The conditions for starting a sequence are defined by StartOfSequenceTriggerSource.

When StartOfSequenceTriggerSource is set to Immediate (default setting), the Sequence Manager doesn't require any further action to allow the Cycle Manager and the Cycle Trigger Manager to proceed with the first cycle.

Depending on the CycleTriggerSource settings of the Cycle Manager the first cycle will be executed:

- Immediately when CycleTriggerSource is set to Immediate
- On execution of the StartCycle command when CycleTriggerSource is set to StartCycle or
- On execution of the StartCycle command or when an event occurs on the I/O toolbox event source designated by CycleTriggerSource.

When StartOfSequenceTriggerSource is set to StartSequence, the Sequence Manager waits for the execution of a StartSequence command before allowing the Cycle Manager and the Cycle Trigger Manager to proceed with the first cycle.

When StartOfSequenceTriggerSource is set to <any-event-source>, the Sequence Manager waits for the execution of a StartSequence command or the occurrence of an I/O toolbox event on the designated event source before allowing the Cycle Manager and the Cycle Trigger Manager to proceed with the first cycle.

Stopping a sequence

The conditions for stopping a sequence are defined by EndOfSequenceTriggerSource.

When EndOfSequenceTriggerSource is set to StopSequence (default setting), the Sequence Manager stops the sequence at the next cycle boundary after the execution of a StopSequence command.

When EndOfSequenceTriggerSource is set to SequenceLength, the Sequence Manager stops automatically the sequence after having executed a number of camera cycles specified by SequenceLength. The sequence can be stopped anticipatively on execution of the StopSequence command. The default `SequenceLength` value is 1; any value up to 16,777,215 is allowed.

When EndOfSequenceTriggerSource is set to *<any-event-source>*, the Sequence Manager waits for the execution of a StopSequence command or the occurrence of an I/O toolbox event on the designated event source before stopping the sequence at the next cycle boundary.

> **NOTE**
> Any combination of StartOfSequenceTriggerSource and EndOfSequenceTriggerSource settings is allowed.

Changing sequence length while camera Is grabbing

Starting with release 10.5, if SequenceLength is changed between start-of-sequence and end-of-sequence events, the new value will be effective for the subsequent sequence.

> **NOTE**
> The value of SequenceLength is latched at the start-of-sequence event.

7.6. CIC Output Signals Routing

Camera Trigger Signal

The frame grabber controls the camera cycle of an asynchronous reset camera by means of the Camera Trigger signal.

The signal can be transmitted from the Coaxlink card to the camera using one of the following media:

- The I/O channel of the CoaXPress link.
- A dedicated wiring driven by a TTL I/O

CoaXPress I/O channel

The Camera Trigger signal is transmitted to the camera as a high-priority Host to Device Trigger message on the CoaXPress I/O channel

See also: "CoaXPress Host To Device Trigger" on page 39for detailed information about the Host to Device Trigger transmitter.

TTL I/O Line

Any TTL I/O line can be configured as a Camera Trigger output. The polarity control of the I/O control block provides an individual polarity control for each I/O port. The mode control of the I/O control block of TTLIO lines provides an individual output driver configuration.

Strobe Signal

I/O Line

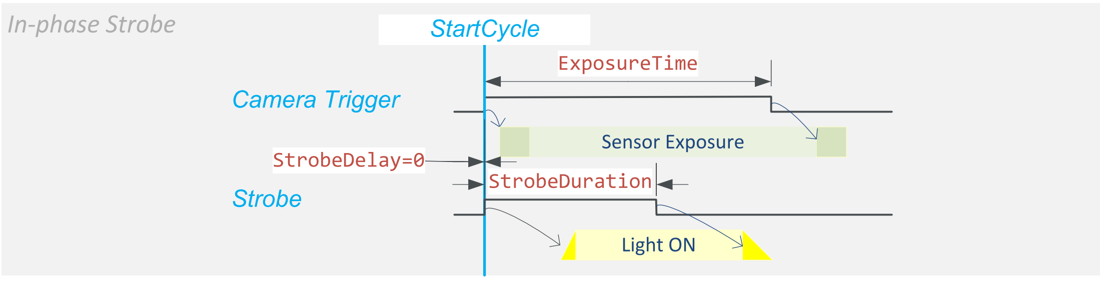
Every output capable I/O line can be configured as an Illumination Strobe output. The polarity control of the I/O control block provides an individual polarity control for each I/O port. The mode control of the I/O control block of TTLIO lines provides an individual output driver configuration for each I/O port used as strobe output.

7.7. CIC Timing Diagrams

Cycle Timing Diagrams

This topic shows timing diagrams of individual CIC cycles and illustrates the ExposureTime, StrobeDuration and StrobeDelay features for 3 use cases corresponding to positive, zero and negative values of StrobeDelay.

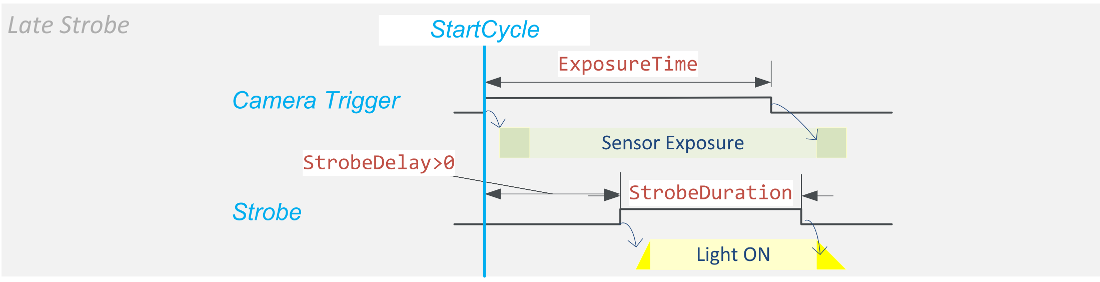
In-phase Strobe Case



StrobeDelay = 0

The Camera Trigger and the Strobe signals goes high immediately after theStartCycle event .

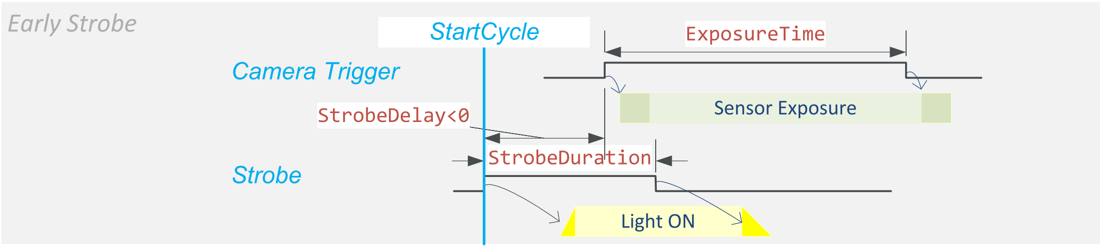
Late Strobe Case



StrobeDelay > 0

The Camera Trigger signal goes high immediately after the StartCycle event and the Strobe signal goes high after StrobeDelay microseconds.

Early Strobe Case



StrobeDelay < 0

The Strobe signal goes high immediately after the StartCycle event and the Camera Trigger signal goes high after a time delay equal to the opposite value of StrobeDelay microseconds.

Consecutive Overlapping CIC Cycles

Introduction

This topic illustrates how the user has to set the ExposureRecovery and TargetFramePeriod features when ExposureReadoutOverlap = True to properly configure the CIC to avoid trigger overruns and maximize the cycle rate.

The Coaxlink driver calculates the duration of the Camera Cycle value from the user-defined settings ExposureTime, ExposureRecovery and TargetFramePeriod by searching the smallest value satisfying the following conditions:

- *Condition 1:* The time interval between consecutive Camera Trigger pulses (*r* – in the drawings) must be greater or equal to the ExposureRecovery settings. This ensures that the Camera Trigger properly flows through the trigger transmission link. It ensures also that a new exposure doesn't begin before the completion of the previous one.
- *Condition 2:* The CIC Cycle duration (a in the drawings) must be big enough to ensure that a new readout doesn't begin before the completion of the previous one.
- *Condition 3:* The CIC Cycle duration must be big enough to include both transitions of the Camera Trigger and the Strobe Signal.

The "readout-limited" use cases illustrate situations where the cycle period is equal to the duration of the readout phase.

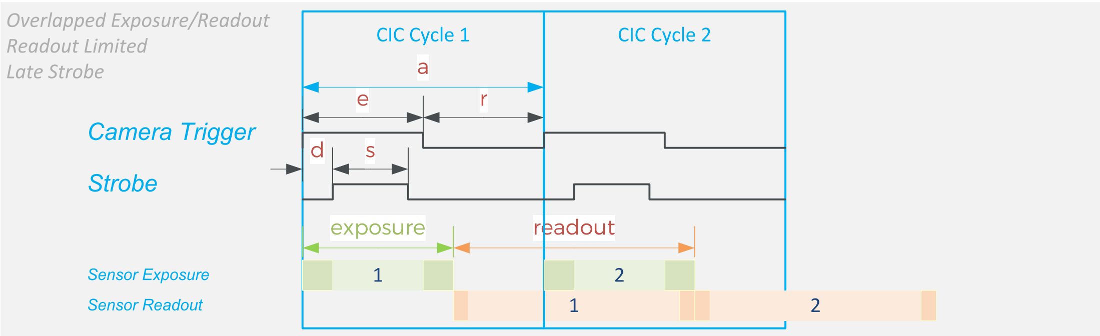
The "exposure-limited" use cases illustrate situations where the cycle period is equal to the duration of the exposure phase.

In the following timing diagrams:

e is the Camera Trigger pulse width. This value is defined by ExposureTime.

d is the Strobe delay. This value is defined by StrobeDelay.

Case 1: Readout Limited (Late Strobe)



The camera cycle rate is only limited by the camera readout time

This situation occurs when `e` (= `Exposure Time`) is significantly smaller than the readout duration.

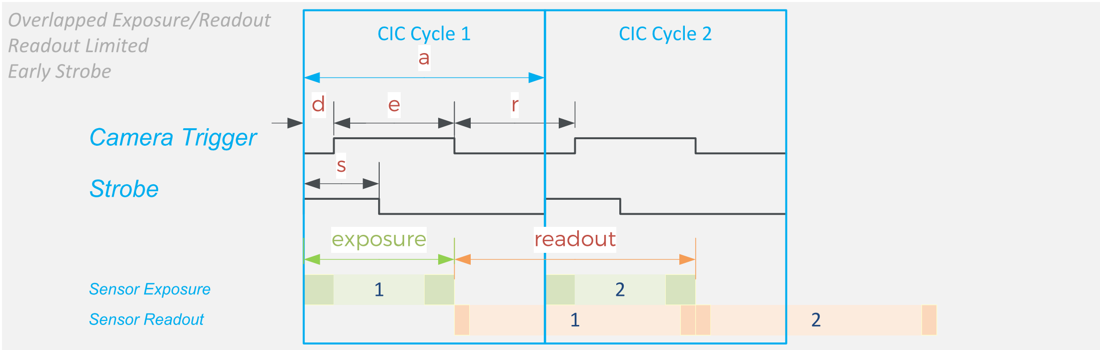
In that situation:

- *r* is likely larger that ExposureRecovery: *Condition 1* is fulfilled.
- The strobe pulse being "inside" the Camera Trigger pulse: *Condition 3* becomes irrelevant when *Condition 1* is fulfilled.
- The *Condition 2* is the only condition used by the Coaxlink driver to calculate the cycle duration.

The optimal duration of the Cycle is equal to the effective duration of the sensor readout phase. This is obtained when the user sets TargetFramePeriod to a value corresponding to the readout duration.

> **NOTE**
> The readout duration can be derived from the maximum frame rate specification of the camera data sheet or experimentally.

Case 2: Readout Limited (Early Strobe)



The camera cycle rate is only limited by the camera readout time (despite the early strobe)

This situation is similar to the case 1. It shows that despite a early strobe, it is possible to reach the maximum cycle rate of the camera.

This situation occurs when *e* (= ExposureTime) is significantly smaller than the readout duration.

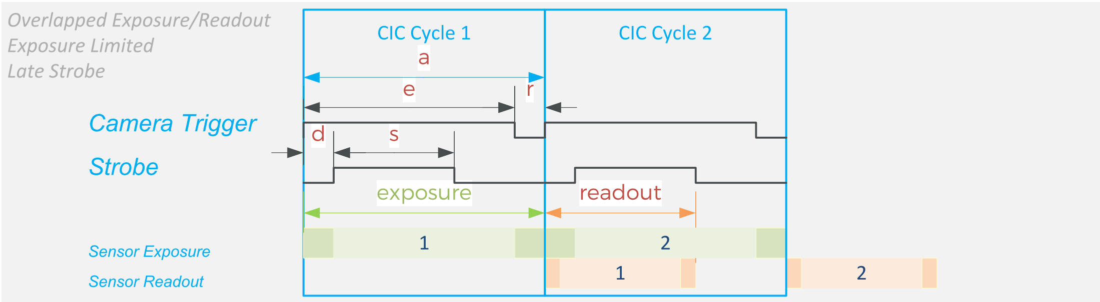
In that situation:

- *r* is likely larger that ExposureRecovery: *Condition 1* is fulfilled.
- The strobe pulse being terminating before the Camera Trigger pulse: *Condition 3* is fulfilled if *r* is greater than *d*. This is the case when (*d* + *e* < readout duration).
- The *Condition 2* is the only condition used by the Coaxlink driver to calculate the cycle duration.

The optimal duration of the Cycle is equal to the effective duration of the sensor readout phase. This is obtained when the user sets TargetFramePeriod to a value corresponding to the readout duration.

NOTE
The readout duration can be derived from the maximum frame rate specification of the camera data sheet or experimentally.

Case 3: Exposure Limited (Late Strobe)



The camera cycle rate is limited by the exposure time settings

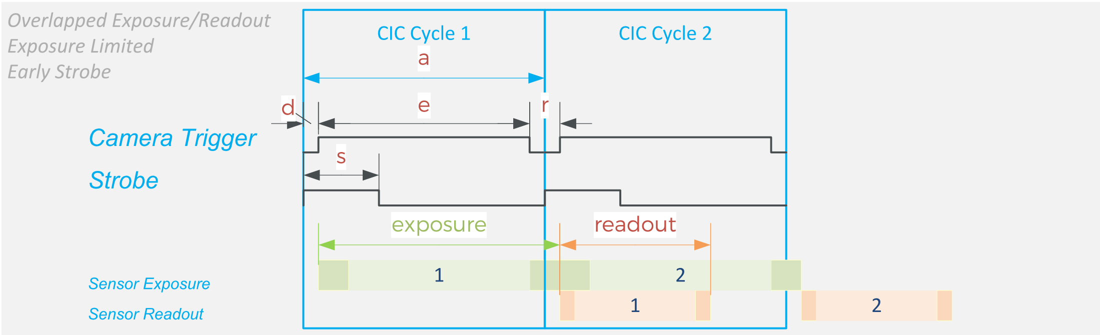
This situation occurs when *e* (= Exposure Time) is significantly larger than the readout duration.

In that situation:

- All cycles being identical, having the readout duration smaller than the exposure duration, implies that *Condition 2* becomes irrelevant.
- The strobe pulse being "inside" the Camera Trigger pulse: *Condition 3* becomes irrelevant when *Condition 1* is fulfilled.
- The *Condition 1* is the only condition used by the Coaxlink driver to calculate the cycle duration .

The optimal duration of the Cycle is equal to the effective duration of the exposure phase. This is obtained when the user sets ExposeRecovery to a value corresponding to the minimal time interval allowed by the camera between consecutive Camera Trigger pulses.

Case 4: Exposure Limited with Early-Strobe



The camera cycle rate is limited by the exposure time settings (despite the early strobe)

This situation is similar to the case 3. It shows that despite an early strobe, it is possible to reach the same cycle rate win case of small negative StrobeDelay values.

This situation occurs when e (= ExposureTime) is significantly larger than the readout duration.

In that situation:

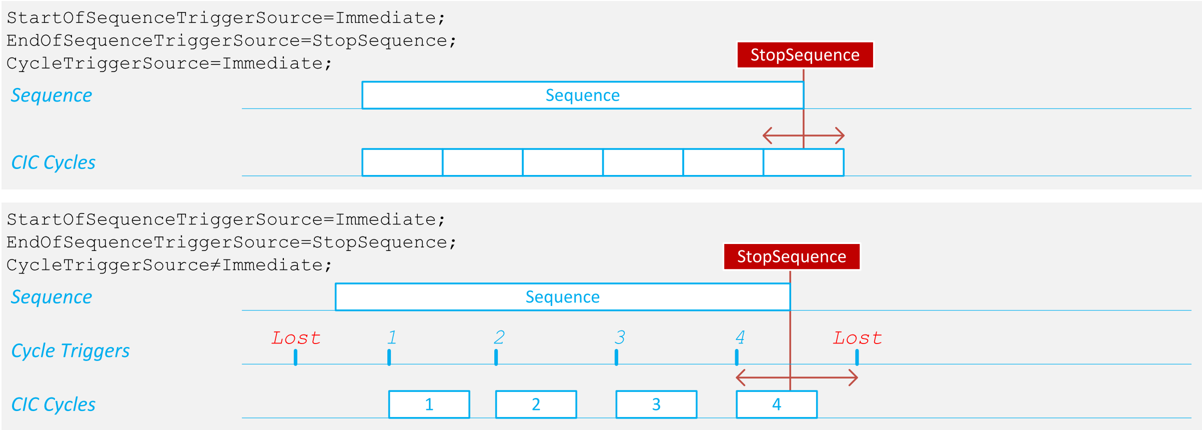
- All cycles being identical, having the readout duration smaller than the exposure duration implies that *Condition 2* becomes irrelevant.
- The strobe pulse terminating before the Camera Trigger pulse: *Condition 3* becomes irrelevant when *Condition 1* is fulfilled and *d* < *r*.
- *Condition 3* and *Condition 1* are the only condition used by the Coaxlink driver to calculate the cycle duration.

The user must set ExposeRecovery to a value corresponding to the largest of of the following two values:

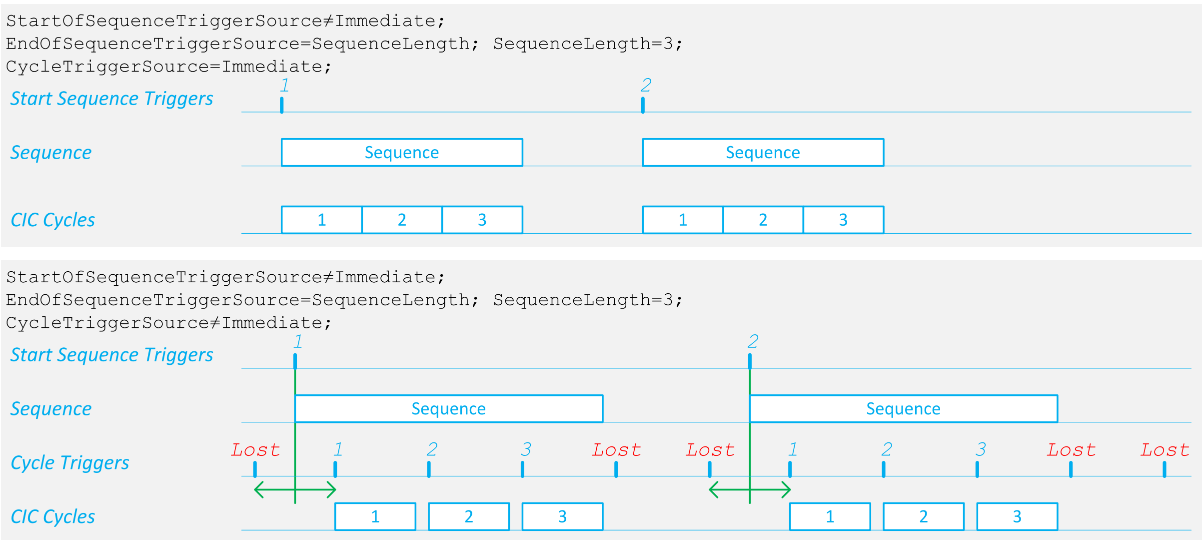
- Minimal time interval allowed by the camera between consecutive Camera Trigger pulses.
- Opposite value of StrobeDelay

NOTE
When CycleTriggerSource = Immediate, the cycle rate can be lowered to the desired rate by assigning a greater value to TargetFramePeriod.

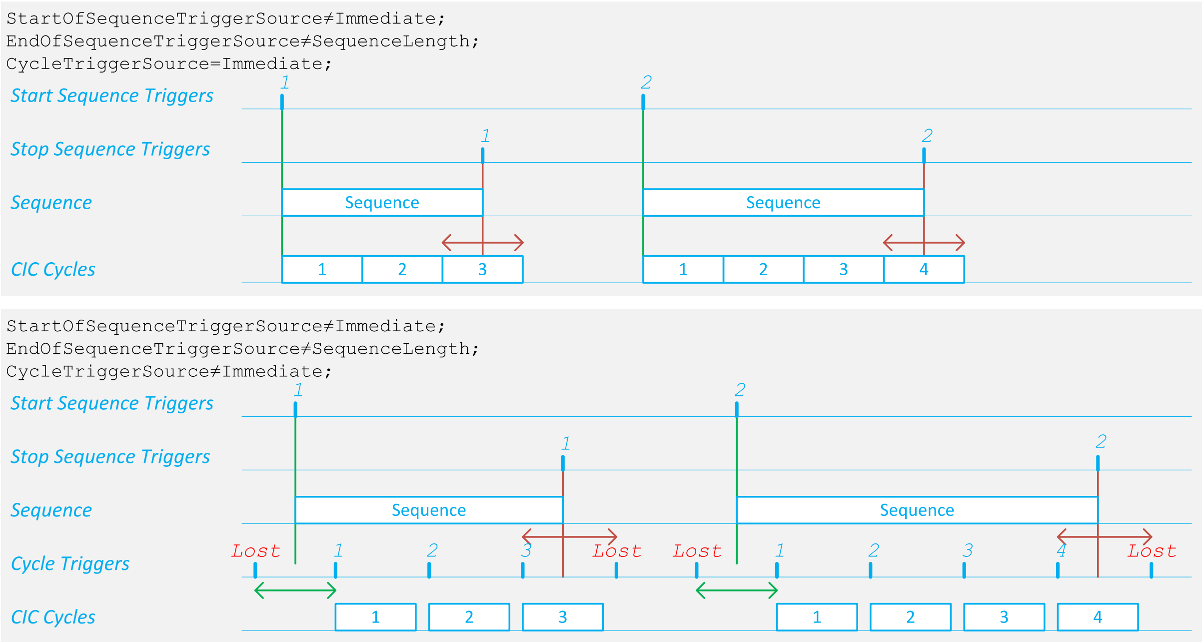
Cycle Sequence Timing Diagrams



Cycle sequences with immediate start



Cycle sequences with triggered start and fixed length



Cycle sequences with triggered start and triggered end

8. General Purpose I/O

8.1. I/O Lines Overview

I/O lines per product

Product	Standard I/O sets		I/O extension
	#1	#2	
1629 Coaxlink Duo PCIe/104-EMB	3300	-	
1630 Coaxlink Mono	On-board	-	-
1631 Coaxlink Duo	On-board	On-board	-
1632 Coaxlink Quad	On-board	On-board	-
1633 Coaxlink Quad G3	On-board	On-board	-
1633-LH Coaxlink Quad G3 LH	On-board	On-board	-
1634 Coaxlink Duo PCIe/104-MIL	3300	-	
1635 Coaxlink Quad G3 DF	On-board	-	-
1637 Coaxlink Quad 3D-LLE	On-board	On-board	-
3602 Coaxlink Octo	On-board	3614	✓
3603 Coaxlink Quad CXP-12	On-board	On-board	✓
3620 Coaxlink Quad CXP-12 JPEG	On-board	On-board	✓
3621-LH Coaxlink Mono CXP-12 LH	On-board	3614	✓
3622 Coaxlink Duo CXP-12	On-board	3614	✓

I/O extension modules: 3610 3612

Standard I/O sets

Standard I/O set #1

I/O Line Type	I/O Line Names	Count
Isolated input	`IIN11`, `IIN12`, `IIN13`, `IIN14`	4
Isolated output	`IOUT11`, `IOUT12`	2
RS-422 input	`DIN11`, `DIN12`	2
TTL I/O	`TTLIO11`, `TTLIO12`	2
Total		**10**

Standard I/O set #2

I/O Line Type	I/O Line Names	Count
Isolated input	IIN21, IIN22, IIN23, IIN24	4
Isolated output	IOUT21, IOUT22	2
LVTTL input/output	TTLIO21, TTLIO22	2
RS-422 input	DIN21, DIN22	2
Total		**10**

I/O and I/O extension modules

3300 I/O module

The **3300 HD26F I/O module for Coaxlink Duo PCIe/104** provides the "Standard I/O set #1" on the previous page.

See also: 3300/3302 Accessories for Coaxlink Duo PCIe/104 in the Coaxlink PCIe/104 hardware manual for more information.

3610/3612 I/O extension modules

The **3610 HD26F I/O Extension Module TTL-RS422** and the **3612 HD26F I/O Extension Module TTL-CMOS5V-RS422** provide the "3610/3612 Module" I/O set:

I/O Line Type	I/O Line Names	Count
RS-422 input	MIO1, MIO3...MIO19	0 to 10
RS-422 output	MIO1, MIO3...MIO19	0 to 10
TTL input	MIO1, MIO2...MIO20	0 to 20
TTL output (3610) CMOS output (3612)	MIO1, MIO2...MIO20	0 to 20
Total		**10 RS-422 | 0 TTL/CMOS** **...** **0 RS-422 | 20 TTL/CMOS**

See also: 3610/3612 I/O Extension Modules in the hardware manual for more information.

3614 I/O extension module

The **3614 HD26F I/O Extension Module - Standard I/O Set** provides the "Standard I/O set #2" above.

See also: 3614 I/O Extension Module in the hardware manual for more information.

I/O line types

I/O Line Type	Electrical Style	Frequency range	Name Prefix
CMOS output	5 Volt CMOS single-ended output	Up to 1 MHz	MIO
Isolated input	Isolated current-sense input compliant with 5 V, 12 V and 24 V signaling levels	Up to 50 kHz	IIN
Isolated output	Isolated 100 mA / 30 V contact output	Up to 50 kHz	IOUT
RS-422 input	RS-422 differential input	Up to 5 MHz	DIN
		Up to 10 MHz	MIO
RS-422 output	RS-422 differential output	Up to 10 MHz	MIO
TTL I/O	5 V-tolerant LVTTL single-ended input and 3.3 V LVTTL totem-pole single-ended output or 3.3 V open-emitter single-ended output or 3.3 V open-collector single-ended output	Up to 5 MHz	TTLIO
TTL input	5 V-tolerant LVTTL single-ended input	Up to 1 MHz	MIO
TTL output	3.3 V LVTTL single-ended output	Up to 1 MHz	MIO

8.2. I/O Lines Usage

Input lines

The input-capable I/O lines (DIN, TTLIO, IN, MIO) can be used as:

- General purpose input: An input signal whose state can be read or monitored by the host application.
- Motion encoder input: A pair of input signals delivered by a quadrature motion encoder and used for triggering the acquisition from the camera.
- Trigger input: An input signal used to trigger the acquisition from the camera.

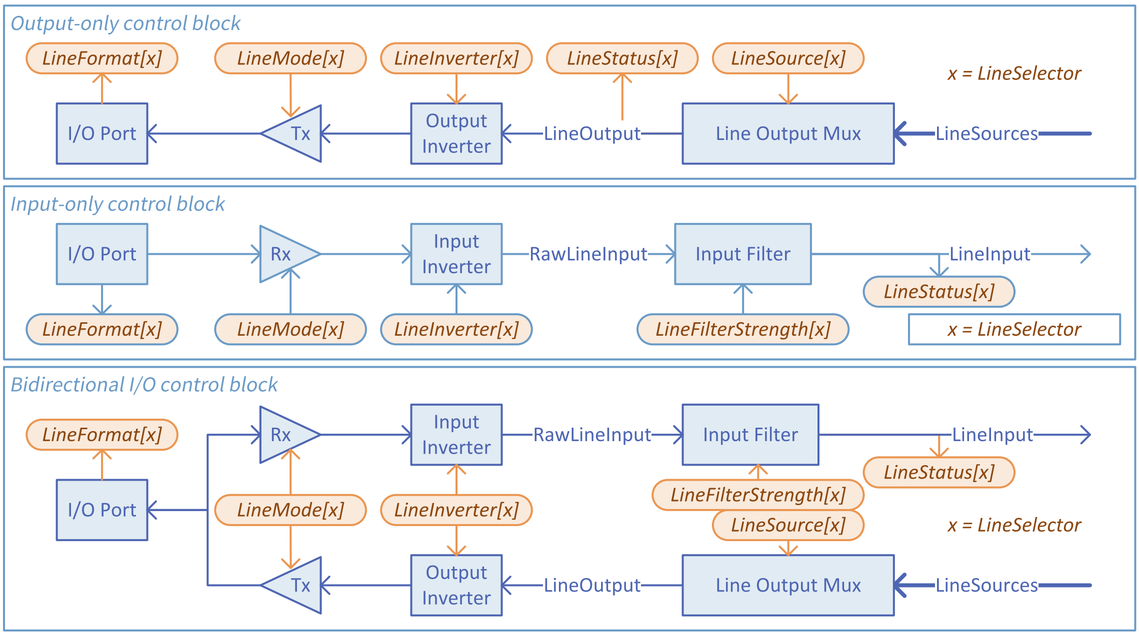
Output lines

The output-capable I/O lines (TTLIO, IOUT, MIO) can be used as:

- General purpose output: An output signal whose state can be set by the host application.
- Strobe output: An output signal usually used to control a strobe light, in synchronization with the camera.
- Camera trigger output (only available on TTLIO): An output signal generated by the Coaxlink card and used to trigger the camera.

8.3. I/O Control Blocks

Every I/O line is controlled through one I/O control block. There a 3 types of control block depending on the input / output capabilities of the I/O line:



In the above figures:

- A thin blue line represents one individual electrical signal path
- A thick blue line represents a collection of electrical signal paths
- The blue arrowhead shows the propagation direction of the electrical signal(s)
- A blue shape represents a functional element of the I/O control block
- An orange (oval) shape represents a GenICam feature; the text inside being the feature name
- An [x] appended to the feature name indicates that the feature is associated with a selector feature (in this case: LineSelector)
- The orange arrowhead indicates the access-mode of the feature: read-only features having incoming arrows, writable features having an outgoing arrow.

Input path

Input-only and bidirectional I/O lines share a common input path structure including:

- The I/O port block representing the I/O pins on the I/O connector(s)
- The Rx block representing the line receiver circuit
- The Input Inverter block representing the user-configurable logic inverter
- The Input Filter block representing the user-configurable glitch-removal filter

Output path

Output-only and bidirectional I/O lines share a common output path structure including:

- The I/O port block representing the I/O pins on the I/O connector(s)
- The Tx block representing the line driver circuit
- The Output Inverter block representing the user-configurable logic inverter
- The Line Output Mux block representing the user-configurable source multiplexer

8.4. Line Format and Line Mode Controls

Introduction

The following tables summarize the details of the I/O Control blocks of each I/O line:

- The first column indicates the LineSelector value
- The second column indicates the bit position in the integer value reported by LineStatusAll
- The third column indicates the value reported by LineFormat
- The fourth column indicates the values that can be assigned to LineMode

Standard I/O sets

Standard I/O set #1

LineSelector	Bit#	LineFormat	LineMode
DIN11	0	DIFF	Input
DIN12	1	DIFF	Input
IIN11	4	ISO	Input
IIN12	5	ISO	Input
IIN13	6	ISO	Input
IIN14	7	ISO	Input
IOUT11	12	ISO	Output
IOUT12	13	ISO	Output
TTLIO11	16	TTL	Input, Output, DriveLow or DriveHigh
TTLIO12	17	TTL	Input, Output, DriveLow or DriveHigh

Standard I/O set #2

LineSelector	Bit#	LineFormat	LineMode
DIN21	1	DIFF	Input
DIN22	3	DIFF	Input
IIN21	8	ISO	Input
IIN22	9	ISO	Input
IIN23	1	ISO	Input
IIN24	11	ISO	Input
IOUT21	14	ISO	Output
IOUT22	15	ISO	Output
TTLIO21	18	TTL	Input, Output, DriveLow or DriveHigh
TTLIO22	19	TTL	Input, Output, DriveLow or DriveHigh

TTLIO ports mode control

The LineMode feature controls the direction and the line driver mode of each individual TTLIO port. Four modes can be selected at any time:

- Input: input only, totem-pole driver disabled (defaut power-up settings),
- Output: totem-pole driver capable of driving low an high,
- DriveLow: open-collector driver capable of driving low only,
- Drivehigh: open-emitter driver capable of driving high only.

NOTE
The two latest configurations allow wired-AND configurations. The line state can be read back through the input port.

I/O modules

3610/3612 I/O Extension modules

LineSelector	Bit#	LineFormat	LineMode
MIO1	20	TTL or DIFF	Input or Output
MIO2	21	TTL	Input or Output
:	:	:	:
MIO19	38	TTL or DIFF	Input or Output
MIO20	39	TTL	Input or Output

MIO format and mode controls

The LineFormat feature controls the electrical style of the MIO ports. Possible values:

- TTL: single-ended (TTL or CMOS)
- DIFF: differential (RS422)

The LineMode feature controls the direction of the MIO ports. Possible values::

- Input: input only,
- Output: totem-pole driver capable of driving low an high.

NOTE
The controls can only be changed during the module configuration .

See also: 3610/3612 I/O Extension Modules for an extensive description of the configuration.

8.5. Line Polarity Control

All the I/O lines are fitted with a polarity control. For bidirectional I/O lines, a single control affects equally both paths.

The line polarity is user-configurable through the LineInverter control.

> **NOTE**
> The user is invited to set the polarity control according to the polarity of the external signal in such a way that the *Line Input* signals of the input path and the *LineSources* signals of the output path are always using positive logic.

8.6. Filter Control

All the I/O input lines are fitted with a glitch-removal filter. The filter strength is user-configurable through the LineFilterStrength control.

The strength control provides 5 positions from 'Lowest' to 'Highest'. The default position is 'Low'.

Each position corresponds to a specific filter time constant for each of the 3 I/O input line types.

Line Filter Time Constant per I/O input line type and LineFilterStrength :

LineFilterStrength	Differential inputs (DIN)	TTL inputs (TTLIO)	Isolated Inputs (IIN)
Lowest	50 ns	50 ns	500 ns
Low	100 ns	100 ns	1 µs
Medium	200 ns	200 ns	2 µs
High	500 ns	500 ns	5 µs
Highest	1 µs	1 µs	10 µs

The user is invited to set the filter strength according to the quality of the external signal. Select a filter strength such that its time constant is:

- Greater than the longest glitch duration
- Greater than the 10%~90% rise/fall time of the signal
- At least 2 times smaller than the smallest signal pulse duration

The glitch removal filter introduces a latency into the input signal path. The latency is equal to the filter time constant when the incoming signal has clean transitions. The latency may increase significantly in case of bad quality signals.

8.7. Line Source Selection

Any output-capable I/O lines is fitted with a source signal multiplexer.

The source signal multiplexers implement a fully-populated signal routing matrix allowing a selection of internal signals to be routed to any output lines.

Selecting a signal source

To select an internal signal to be used as signal source for the line driver of an output capable I/O line:

Step 1: Select an I/O line by assigning the appropriate value to the LineSelector feature

Step 2: Assign the appropriate value to the SignalSource feature

Output-lines routing matrix

Source Signal	Possible destinations
Any Bit of the User Output Register	Any output line
Any Device Module Camera Trigger Output	Any TTLIO output line
Any Device Module Strobe Output	Any output line
Steady Low	Any output line

8.8. Logical I/O Line State

Logical I/O line state

The (logical) state of an I/O line is the logical state of an electrical signal of the I/O control block:

- For input-capable I/O lines: the *LineInput* signal: a point in the input path of the I/O control block that is located after the Input Inverter.
- For output-only I/O lines: the *LineOutput* signal, a point in the output path of the I/O control block that is located before the Output Inverter.

Getting the state of a single I/O line

1. *Step 1:* Select an I/O line by assigning the appropriate value to the LineSelector feature
2. *Step 2:* Obtain directly the line status by getting the value of the LineStatus feature

Getting the state of all I/O lines in a single operation

Get the value of the LineStatusAll feature.

Each bit of the integer corresponds to an I/O line. A bit at one corresponds to a line logical state being high.

8.9. Physical I/O Line State

The physical state of the I/O line state does not only depend on the value reported when reading `LineStatus` but also on the following I/O block settings: `LineFormat`, `LineMode`, and `LineInverter`

LineFormat	LineMode	LineInverter/LineStatus	Physical I/O line state
DIFF	Input	False/False or True/True	*(VIN+ - VIN-) < VThreshold*
		False/True or True/False	*(VIN+ - VIN-) > VThreshold or unconnected I/O line*
ISO	Input	False/False or True/True	Opto coupler is OFF. The line current is < 1 mA. *Line may be left unconnected or connected with the wrong polarity.*
		False/True or True/False	Opto coupler is ON. The line current is > 1 mA.
ISO	Output	False/False or True/True	Opto coupler is OFF
		False/True or True/False	Opto coupler is ON
TTL	Input	False/False or True/True	The line voltage is < 0.8 Volt.
		False/True or True/False	The line voltage is > 2.0 Volt.
TTL	Output	False/False or True/True	The line is driven LOW.
		False/True or True/False	The line is driven HIGH.
TTL	DriveLow	False/False or True/True	The line is driven LOW.
		False/True or True/False	The line voltage is > 2.0 Volt.
TTL	DriveHigh	False/False or True/True	The line voltage is < 0.8 Volt.
		False/True or True/False	The line is driven HIGH.

8.10. Line Driver Physical Output States

The line driver output state depends not only on the logical level of the selected Line Output signal but also on the following I/O block settings: `LineFormat`, `LineMode`, and `LineInverter`.

LineFormat	LineMode	LineInverter / LineOutput logical level	Line driver output state
ISO	Output	False / low or True / high	The opto-coupler switch is turned OFF.
		False / high or True / low	The opto-coupler switch is turned ON.
TTL CMOS	Output	False / low or True / high	The I/O line is driven LOW.
		False / high or True / low	The I/O line is driven HIGH.
TTL	DriveLow	False / low or True / high	The I/O line is driven LOW.
		False / high or True / low	The I/O line is not driven.
TTL	DriveHigh	False / low or True / high	The I/O line is not driven.
		False / high or True / low	The I/O line is driven HIGH.

8.11. Initial States

At power-on, the I/O lines are in the following state:

Differential Inputs

- LineInverter = False,
- LineFilterStrength = Low.

TTL Inputs/outputs

- LineMode = Input (The line is not driven),
- LineInverter = False,
- LineFilterStrength = Low.

Isolated Inputs

- LineInverter = False,
- LineFilterStrength = Low.

Isolated Outputs

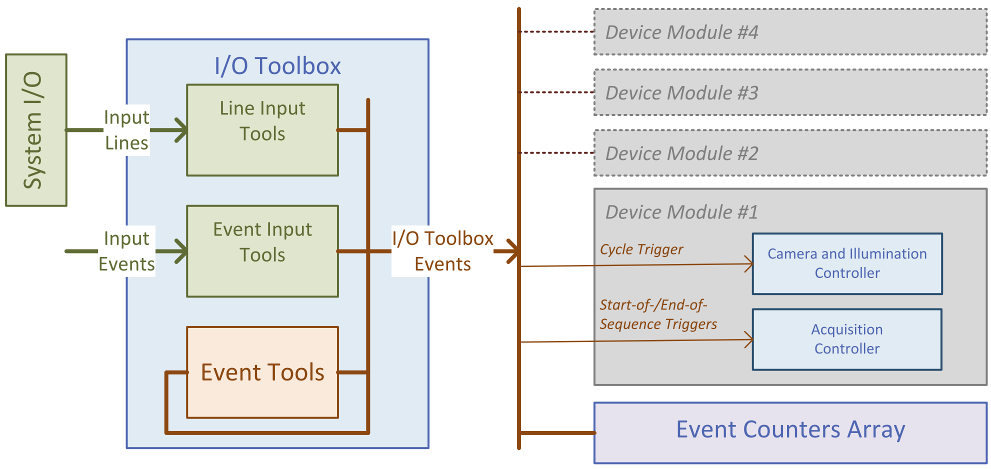
- LineInverter = False,
- LineFilterStrength = Low,
- The opto-coupler is OFF.

9. I/O Toolbox

9.1. Introducing the I/O Toolbox

The **I/O Toolbox** is a configurable array of tools that builds-up streams of event pulses from external signals applied to the input-capable I/O lines.

I/O Toolbox context



I/O Toolbox context block diagram

There is only one I/O Toolbox instance per Coaxlink card. It belongs to the GenTL Interface module.

The input-capable I/O lines (*Input Lines*) and a selected event sources (*Input Events*) are feeding the *Input Tools* the I/O Toolbox.

All the I/O Toolbox tools generate one (or more) event streams (*I/O Toolbox Events*).

The I/O Toolbox Events streams are distributed to various consumers on the card.

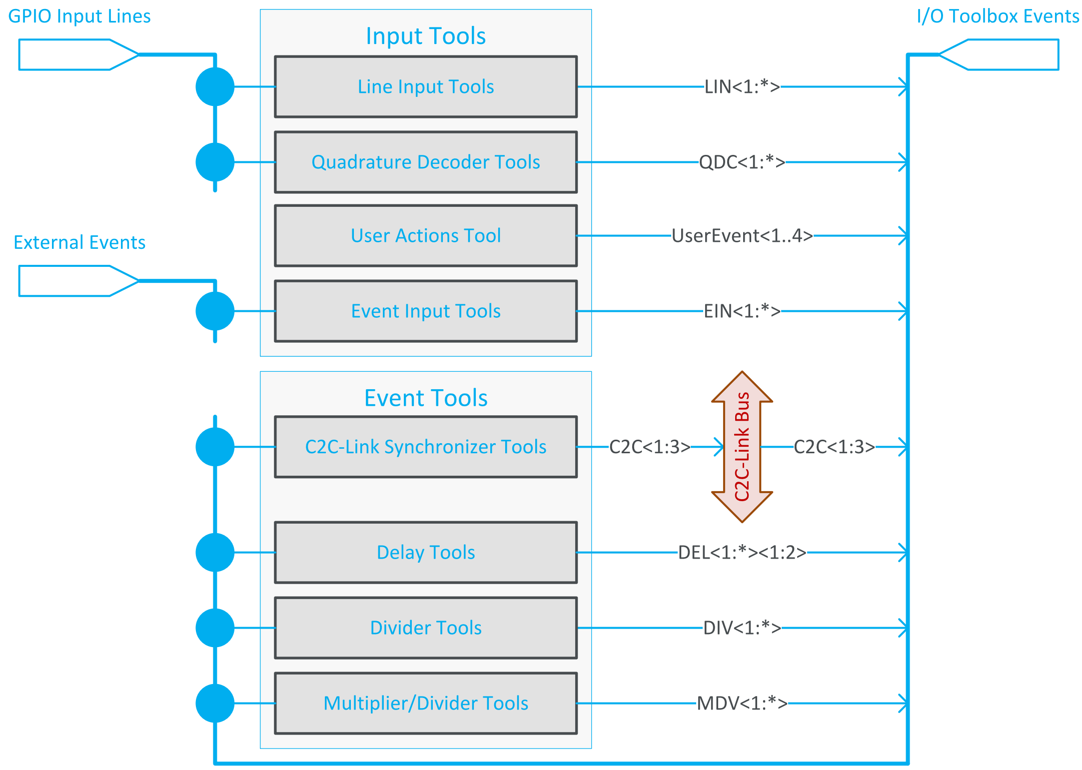
Any I/O Toolbox events can be used in any GenTL Device module of the card as:

- The hardware Cycle Trigger source.
- The hardware Start-of-scan Trigger source.
- The hardware End-of-scan Trigger source.

Every I/O Toolbox event stream is associated with a 32-bit event counter.

I/O Toolbox structure

The **I/O Toolbox** is a configurable array of tools that builds-up streams of event pulses from external signals applied to the input-capable I/O lines.



I/O Toolbox structure diagram

I/O Toolbox input tools

The I/O Toolbox input tools are fed by the input-capable GPIO lines and a selected set of hardware event sources from outside the I/O Toolbox.

They deliver one I/O Toolbox event stream.

There are four types of input tools:

1. The *Line Input Tools*, for use with sensors and detectors using a single GPIO input line.
2. The *Quadrature Decoder Tools* for use with quadrature motion encoders using two GPIO input lines.
3. The *User Actions Tool* for use by the application software to generate user events
4. The *Event Input Tools* for use in line-scan data-forwarding applications.

The *GPIO Input Lines Interconnections Matrix* allows any Line Tool to be fed by any GPIO input line.

The *External Events Interconnections Matrix* allows any Event Input Tool to be fed by the external event source.

NOTE
Currently, only the CoaXPress Host interface GPIO message receiver of connector A is available.

I/O Toolbox event tools

The I/O Toolbox event tools are fed by one (or more) I/O Toolbox event stream(s) and deliver one (or more) I/O Toolbox event stream(s).

There are three types of events tools:

1. The *Divider Tool* generates an event stream by keeping 1 event out of D events of the input stream.
2. The *Multiplier/Divider Tool* generates an event stream having M events every D events of the input stream.
3. The *Delay Tool* delays the events of one (or two) streams, either by a configurable period of time or by a configurable number of motion encoder ticks.

The *Internal Events Interconnection Matrix* allows any Event Tool to be fed by any I/O Toolbox event stream.

Tools can be cascaded to form a tool chain:

A tool chain always begins with a Line Tool.

A (Line or Event) Tool may drive 0, 1 or several Event Tools.

9.2. I/O Toolbox Composition

The composition of the I/O Toolbox is product- and firmware-variant-specific:

1630 Coaxlink Mono

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0

1631 Coaxlink Duo

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0
1-camera, line-scan	8	1	1	1	2	1	0
2-camera	8	2	2	2	2	1	0
2-camera, line-scan	8	2	2	2	2	1	0

1632 Coaxlink Quad

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0
1-camera, line-scan	8	1	1	1	2	1	0
2-camera	8	2	2	2	2	1	0

1633 Coaxlink Quad G3

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0
1-camera, line-scan	8	1	1	1	2	1	0
1-slm-camera	8	1	1	1	2	1	0
1-sls-camera	8	1	1	1	2	1	0
2-camera	8	2	2	2	2	1	0
2-camera, line-scan	8	2	2	2	2	1	0
3-camera	8	2	2	2	2	1	0
4-camera	8	4	4	4	4	1	0
4-camera, line-scan	8	4	4	4	4	1	0
1-camera, 4-data-stream	8	1	1	1	2	1	0

1633-LH Coaxlink Quad G3 LH

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0
1-camera, line-scan	8	1	1	1	2	1	0
1-slm-camera	8	1	1	1	2	1	0
1-sls-camera	8	1	1	1	2	1	0
2-camera	8	2	2	2	2	1	0
2-camera, line-scan	8	2	2	2	2	1	0
3-camera	8	2	2	2	2	1	0
4-camera	8	4	4	4	4	1	0
4-camera, line-scan	8	4	4	4	4	1	0
1-camera, 4-data-stream	8	1	1	1	2	1	0

1629 Coaxlink Duo PCIe/104-EMB

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0
1-camera, line-scan	8	1	1	1	2	1	0
2-camera	8	2	2	2	2	1	0

1634 Coaxlink Duo PCIe/104-MIL

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0
1-camera, line-scan	8	1	1	1	2	1	0
2-camera	8	2	2	2	2	1	0

1635 Coaxlink Quad G3 DF

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0
1-df-camera	8	1	1	1	2	1	0
1-camera, line-scan	8	1	1	1	2	1	0
1-df-camera, line-scan	8	1	1	1	2	1	2

1637 Coaxlink Quad 3D-LLE

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0

3602 Coaxlink Octo

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0
1-camera, line-scan	8	1	1	1	2	1	0
2-camera	8	1	1	1	2	1	0
2-camera, line-scan	8	1	1	1	2	1	0
4-camera	8	1	1	1	2	1	0
4-camera, line-scan	8	4	4	4	4	1	0
5-camera	8	1	1	1	2	1	0
8-camera	8	1	1	1	2	1	0

3603 Coaxlink Quad CXP-12

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0
1-camera, line-scan	8	1	1	1	2	1	0
2-camera	8	1	1	1	2	1	0
4-camera	8	4	4	4	4	1	0

3620 Coaxlink Quad CXP-12 JPEG

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
4-camera	8	4	4	4	4	1	0

3621-LH Coaxlink Mono CXP-12 LH

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0

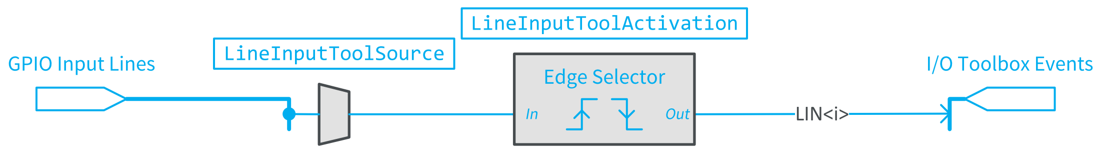
3622 Coaxlink Duo CXP-12

Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	2	1	0
2-camera	8	2	2	2	2	1	0

9.3. Line Input Tool

Tool Name	Short Name	Inputs Count/Type	Outputs Count/Type/Name
Line Input Tool	LIN	1 GPIO input line	1 event stream: LIN<i>

Diagram



LIN tool functional and wiring diagram

Any input-capable GPIO line can be selected as the input source.

The tool feeds one I/O Toolbox event stream named LIN<i>.

Operation

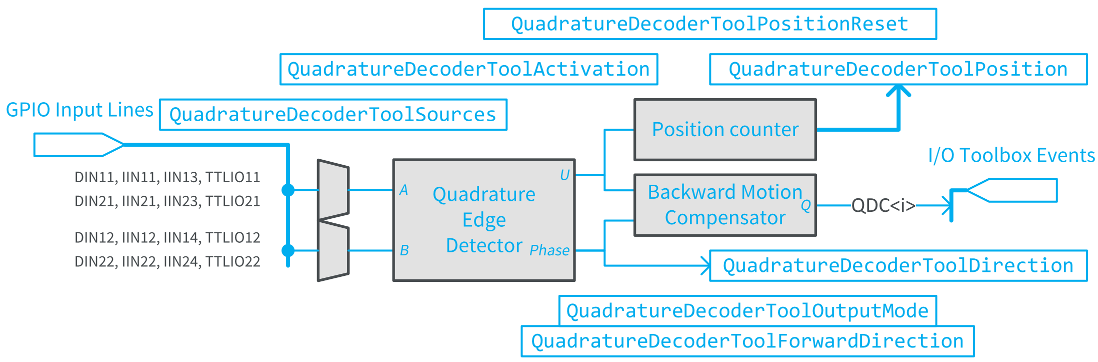
The Line Input tool detects the rising or the falling edge of the LineInput signal delivered by the I/O Control block selected by LineInputToolSource.

The Line Input tool delivers one event at every rising or falling edge or both according to the LineInputToolActivation settings.

9.4. Quadrature Decoder Tool

Tool Name	Short Name	Inputs Count/Type	Outputs Count/Type/Name
Quadrature Decoder Tool	QDC	2 paired I/O input lines	1 event stream: QDC<i> 1 status bit: QDC<i>Direction 1 status word: QDC<i>Position

Diagram



QDC tool functional and wiring diagram

The quadrature edge detector is fed by a pair of signals named A and B delivered by a *phase-quadrature motion encoder* device.

The source selectors allows following pairs of adjacent input-capable GPIO input lines to be selected as A/B input sources: DIN11-DIN12, DIN21-DIN22, TTLIO11-TTLIO12, TTLIO21-TTLIO22, IIN11-IIN12, IIN13-IIN14, IIN21-IIN22, and IIN23-IIN24.

The tool includes the following function blocks:

- A quadrature edge detector
- A backward motion compensator
- A 32-bit position counter

The tool delivers:

- One I/O Toolbox event stream named QDC<i>.
- A *direction* status bit indicating the direction of the motion.
- A *position* status word indicating the position offset.

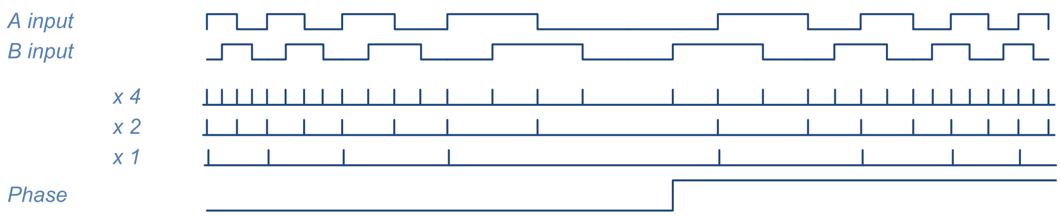
Operation

The Quadrature Decoder Tool decodes the A/B signals and delivers 1, 2, or 4 events every A/B cycle, possibly filtered by the backward motion compensator.

Quadrature edge detector

The *quadrature edge detector* analyzes the transitions on the A/B lines. It delivers:

- An event stream, named U, having 1, 2, or 4 events every A/B cycle according the QuadratureDecoderToolActivation settings
- An identification of the phase between A and B (A leads B or vice-versa)



Quadrature edge detector waveforms

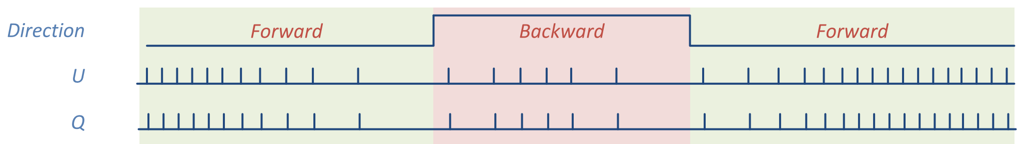
The U stream may be filtered by the backward motion compensator before being delivered to the QDC<i> output.

The phase indication may be inverted according to the QuadratureDecoderToolForwardDirection settings before being delivered to the *Direction* output.

Backward motion compensator

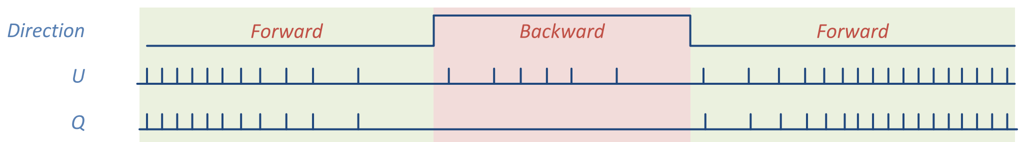
The *backward motion compensator* (BMC) filters the U stream according to the QuadratureDecoderToolOutputMode setting.

When set to Unfiltered, all the events of the U stream are delivered to the QDC<i> output:



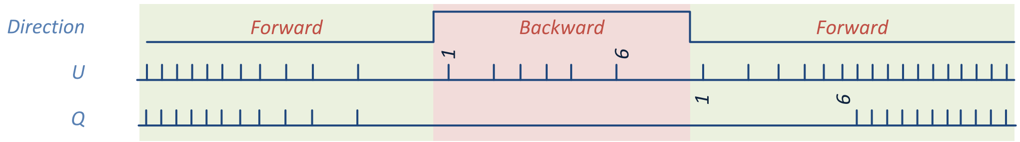
BMC waveforms - Unfiltered

When set to ForwardOnly, only the events corresponding to the forward direction are delivered to the QDC<i> output:



BMC waveforms - Forward Only

When set to FirstPassForwardOnly, only the events corresponding to the first pass in the forward direction are delivered to the QDC<i> output:



BMC waveforms - First Pass Forward Only

Position Counter

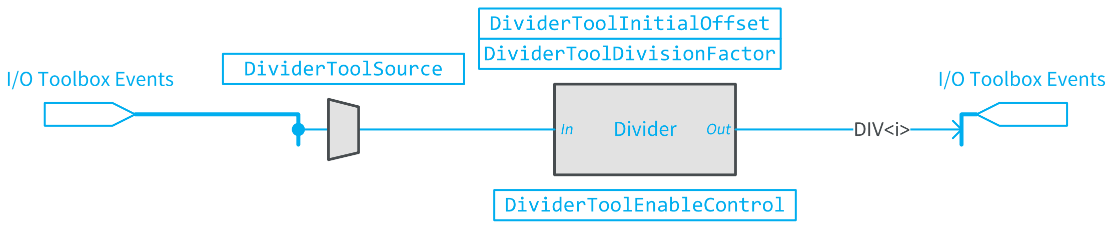
The *position counter* increments by 1 for any U event corresponding to the forward direction and decrements by 1 for the backward direction.

The counter can be reset using the QuadratureDecoderToolPositionReset command.

9.5. Divider Tool

Tool Name	Short Name	Inputs Count/Type	Outputs Count/Type/Name
Divider Tool	DIV	1 Toolbox event stream	1 event stream: DIV<i>

Diagram

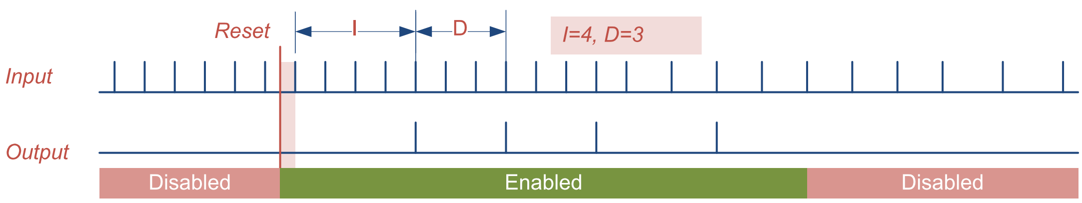


DIV tool functional and wiring diagram

Any I/O Toolbox event stream can be selected as the input source.

The tool feeds one I/O Toolbox event stream named DIV<i>.

Operation



DIV tool waveforms

Once enabled, the Divider tool skips the first – **I** – input events before delivering an event every D input events.

The division factor – **D** – is defined by DividerToolDivisionFactor. The default value is 2 and the value range is 1 … 65535.

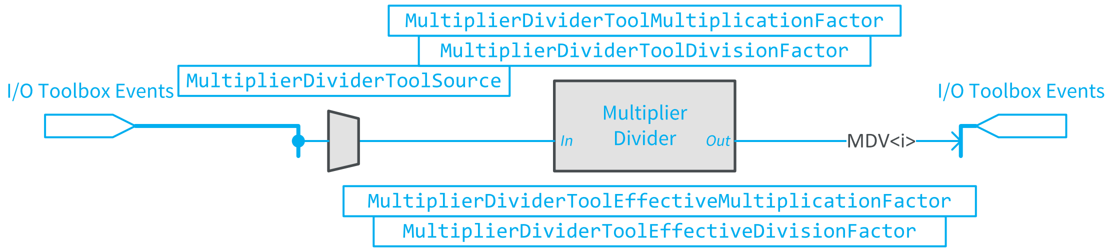
The initial offset – **I** – is defined by DividerToolInitialOffset. The default value is 0 and the value range is 0 … 65535.

The operation state is defined by DividerToolEnableControl. The default value is Disable.

9.6. Multiplier/Divider Tool

Tool Name	Short Name	Inputs Count/Type	Outputs Count/Type/Name
Multiplier/Divider Tool	MDV	1 Toolbox event stream	1 Toolbox event stream: MDV<i>

Multiplier/Divider Tool Wiring Diagram



MDV tool functional and wiring diagram

Any I/O Toolbox event stream can be selected as the input source.

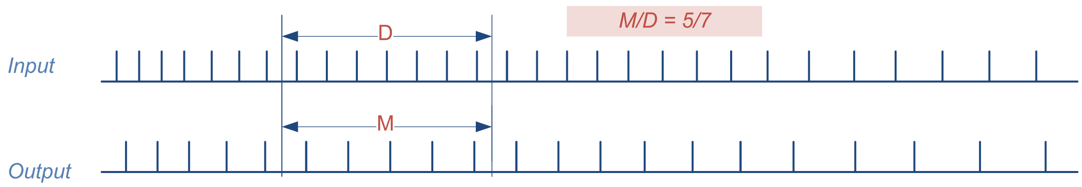
The tool feeds one I/O Toolbox event stream named MDV<i>.

Multiplier/Divider Tool Operation

The Multiplier/Divider tool multiplies and/or divides the input rate by any rate conversion ratio – RCR – value in the range 0.001 to 1000.0.

The Multiplier/Divider tool measures the time interval between every consecutive input events and adapts the output rate accordingly.

The Multiplier/Divider is *frequency accurate*. The output frequency is strictly proportional to the input frequency provided that the input frequency is stable (or varies slowly). In such conditions, the Multiplier/Divider delivers M events for every D input events.



MDV tool waveforms

The Rate Conversion Ratio is configured as the ratio of two float numbers:

- The *M* value is defined by MultiplierDividerToolMultiplicationFactor. The default value is 1.0 and the value range is 0.001 to 1000.0.
- The *D* value is defined by MultiplierDividerToolDivisionFactor. The default value is 1.0 and the value range is 0.001 to 1000.0.

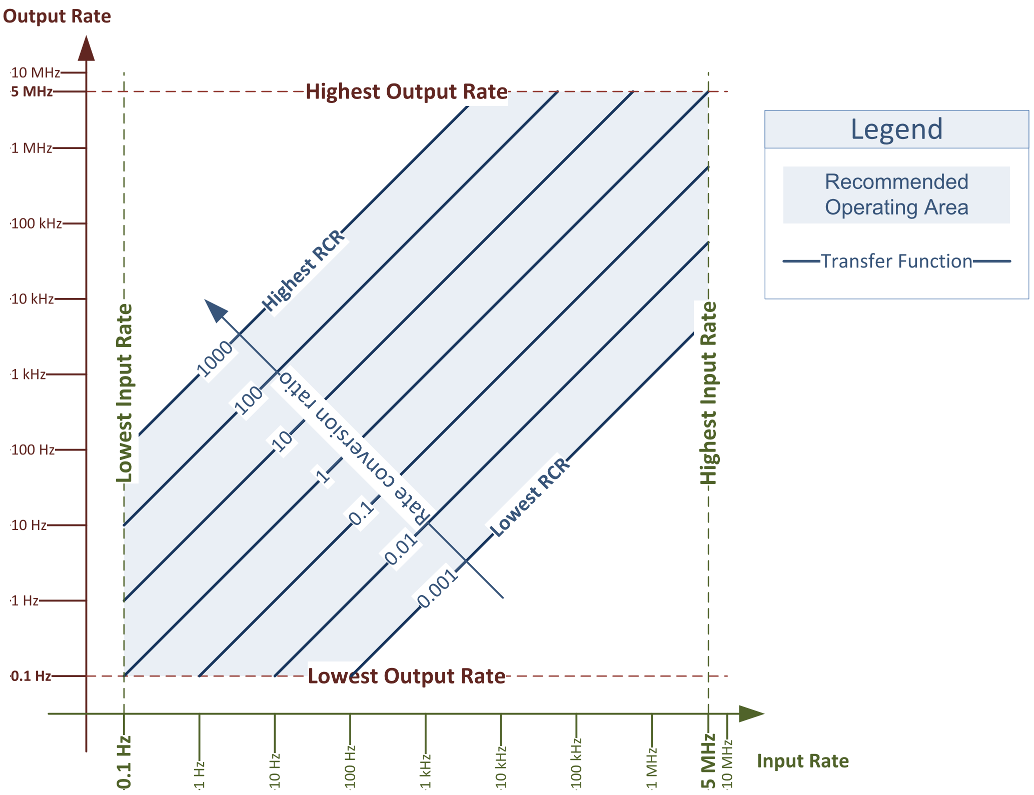
The effective multiplication and division factors are respectively reported by MultiplierDividerToolMultiplicationFactor and MultiplierDividerToolDivisionFactor.

> **NOTE**
> The effective values may slightly differ from the specified values. However, the RCR relative error remains negligible (less than 1/1000).

> **NOTE**
> Frequency variations of the input event stream are reported to the output event stream with a latency of 1 period of the input event stream. Such a latency induces some phase errors in the output event stream. The phase of the output event stream accumulates retards when the input frequency increases and vice-versa when the input frequency increases. Consequently, the Multiplier/Divider is *not phase accurate*.

Multiplier/Divider Tool Operating Limits

Characteristic	Symbol	Min	Max
Rate Conversion Ratio	RCR	0.001	1000
Input Rate	f_{IN}	0.1 Hz	5 MHz
Output Rate	f_{OUT}	0.1 Hz	5 MHz

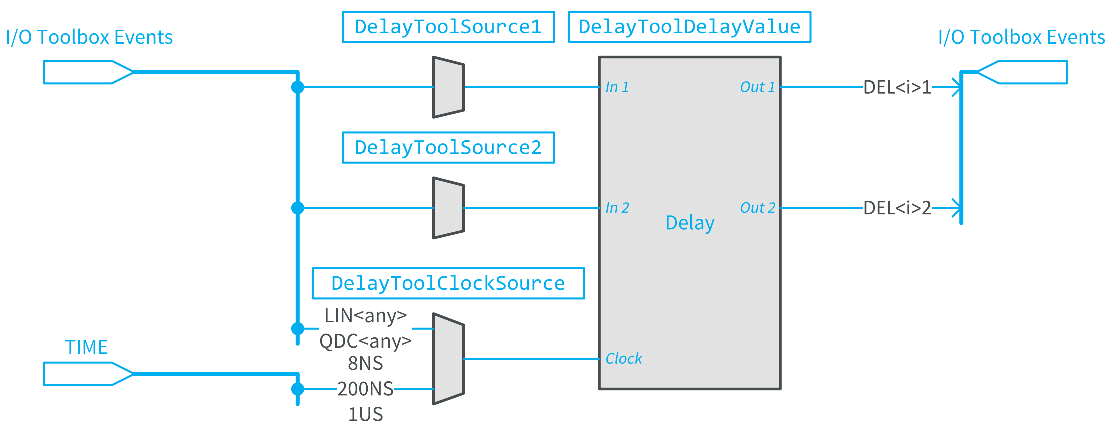


MDV tool operating limits diagram

9.7. Delay Tool

Tool Name	Short Name	Inputs Count/Type	Outputs Count/Type/Name
Delay Tool	DEL	2 Toolbox event streams 1 clock signal	2 Toolbox event stream: DEL<i>1, DEL<i>2

Diagram



DEL tool functional and wiring diagram

Any I/O Toolbox event stream can be selected as the input 1 source.

Any I/O Toolbox event stream can be selected as the input 2 source.

The tool feeds two I/O Toolbox event streams. The outputs of the tool instance <i> are named DEL<i>1 and DEL<i>2.

Operation

The event streams applied on either inputs (In1 and In2) are replicated on the corresponding output (Out1 and Out2) after a configurable number of clock tick events.

The sources are selected by DelayToolSource1 and DelayToolSource2 respectively.

The same delay applies to both channels. The common delay is defined by DelayToolDelayValue.

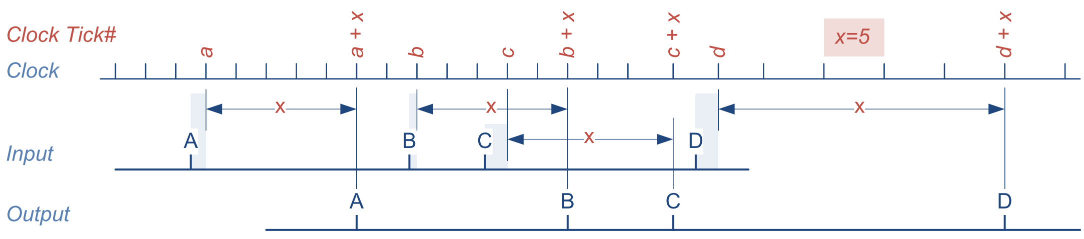
The same clock source applies to both channels. The clock source is defined by DelayToolClockSource. It can be a *time base* or a *line tool event stream*.

Selecting a *time base* implements a time delay function. The available time bases are:

- 8NS: A 125 MHz high accuracy regular time base allowing delays from *40 nanoseconds* up to *134 milliseconds* by steps of 8 nanoseconds
- 200NS: A 5 MHz high accuracy regular time base allowing delays from *200 nanoseconds* up to *3.35 seconds* by steps of 200 nanoseconds
- 1US: A 1 MHz high accuracy regular time base allowing delays from *1 microsecond* up to *16.7 seconds* by steps of 1 microsecond

Selecting a *line tool event stream* implements a position offset function when the line tool is fed by a motion encoder device. Any available Line Input tool or Quadrature Decoder tool can be used as delay clock source. The delay range is *1* up to *16,777,215 events*.

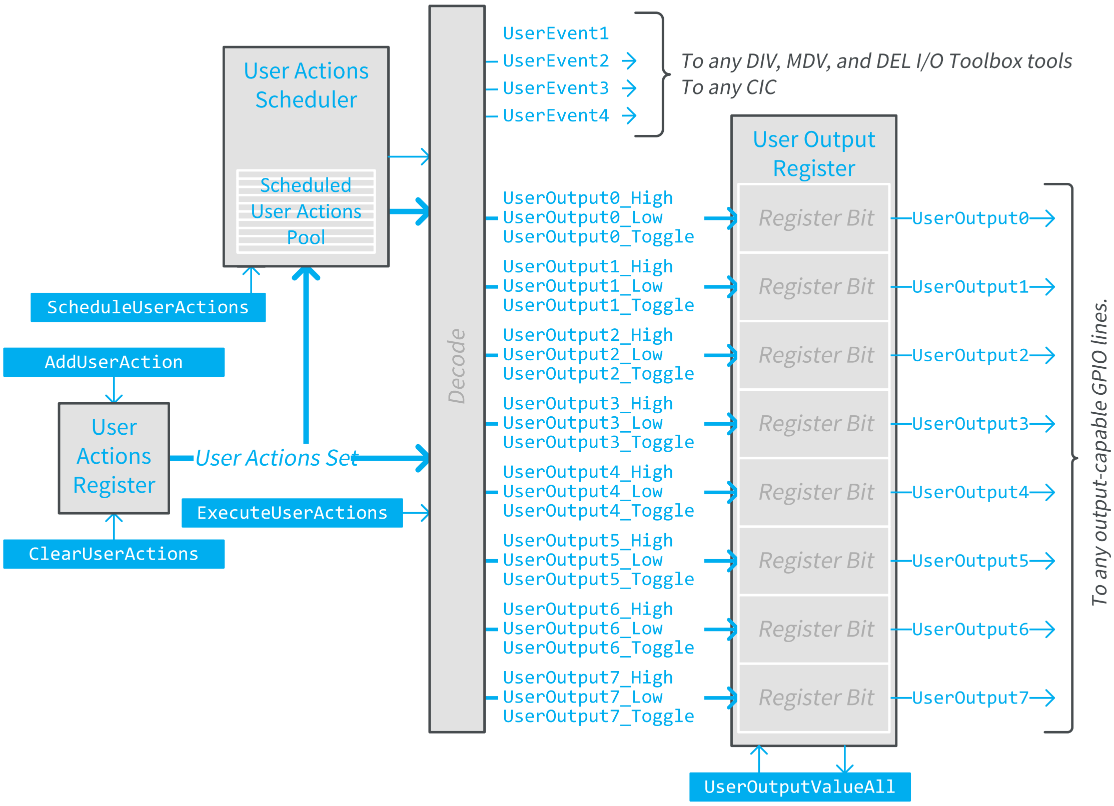
> **WARNING**
> The Delay tool operates as a delay line. The tool may accept a new event while the previous one is not yet delivered! The Delay tool is capable of recording, globally for all channels, up to 16 distinct events.



DEL tool waveforms

9.8. User Actions Tool & User Output Register

Introduction



UAS tool functional and wiring block diagram

The *User Actions Tool* (UAS) allows the application software to perform the following *User Actions*:

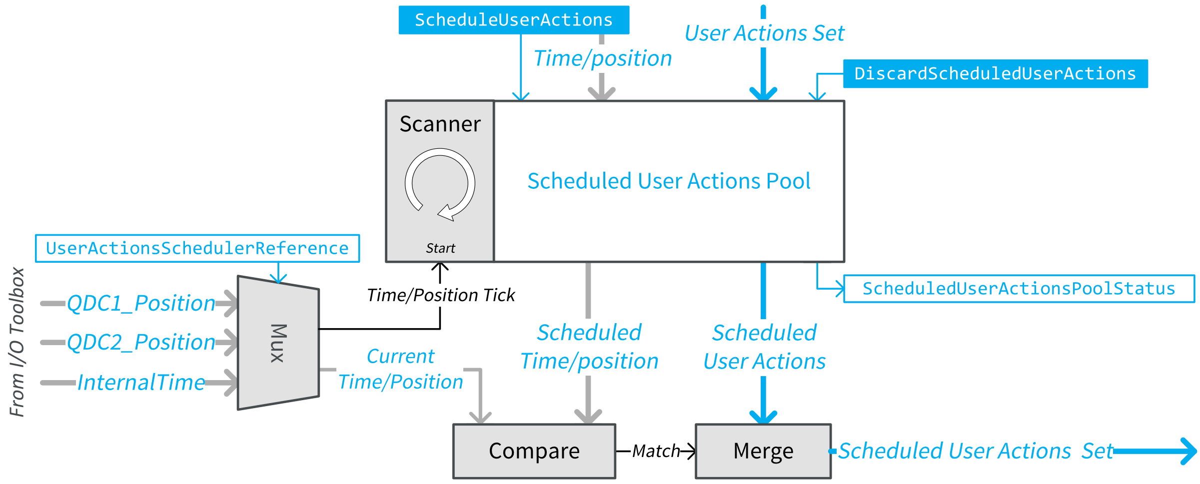
- Generate events on any User Events source.
- Setting high any bit of the User Output Register.
- Setting low any bit of the User Output Register
- Toggling any bit of the User Output Register.

With the ClearUserActions and AddUserActions GenICam features, the user application can define a *User Actions Set* composed with all the actions to be executed simultaneously.

After having defined the User Actions Set, the application software has two options for its execution:

- Using the ExecuteUserActions GenICam feature to execute immediately all the actions defined in the *User Actions Set*.
- Using the ScheduleUserActions GenICam feature to delegate the execution of the User Actions Set to the *User Actions Scheduler*.

User actions scheduler



User actions scheduler functional diagram

The UAS function block allows an application software to postpone the execution of the actions.

Scheduler reference settings

Prior to using the User Actions Scheduler, the user application has to select a 32-bit reference counter. This is achieved by assigning one of the following three values to the UserActionsSchedulerReference GenICam feature:

- InternalTime selects the *Coaxlink card local time*: A monotonic time base that increments by 1 every 1 microsecond and wraps around after about 71 minutes when it reaches the maximum value of 4,294,967,295.
- QDC1Position and QDC2Position selects the *Position Counter* of the Quadrature Decoder tools QDC1 and QDC2 respectively.

WARNING
For correct operation of the UAS with a position reference, the position counter must increment monotonically and not faster than every microsecond.

NOTE
To ensure monotonic increments of the QDC position counter:

- Set properly QuadratureDecoderToolForwardDirection such that the counter increments when the object moves in the forward direction.
- If it exists any backward motion, prevent the position counter to decrement by setting the QuadratureDecoderToolOutputMode to ForwardOnly or to FirstPassForwardOnly.

Scheduled User Actions Pool operation

The Scheduled User Actions Pool is a memory area where the Scheduled User Actions Sets are stored by the user application. The pool has 64 locations.

To add a new Scheduled User Actions to the Pool, the user application must:

1. Ensure that there is at least one free location by getting the value of the ScheduledUserActionsPoolStatus GenICam feature,
2. Define a User Actions Set,
3. Determines the time/position 32-bit value when the actions are to be executed,
4. Set this value to ScheduledUserActions GenICam feature.

Scheduled User Actions are removed from the pool when they are executed.

The pool can be cleared at any time by executing the DiscardScheduledUserActions GenICam command.

Scanner operation and scheduled actions Execution

At every increment of the 32-bit (time or position) reference counter, the Scanner reads all the locations and compares the scheduled reference time/position with the current time/position count value.

When the values are identical, the Scheduled User Actions Set is elected for execution at the end of the scan and removed from the pool.

When multiple sets are elected for execution, their actions are merged.

> **NOTE**
> Merging a set low and a set high action on the same User Output Register bit results into a toggle action.

At the end of the scan: the merged elected actions are executed simultaneously. The time delay from the reference tick up to the execution of the elected actions is very small (sub-microsecond) and constant.

User events sources

There are User Events sources named `UserEvent1` … `UserEvent4`.

Any User Event can be used:

- As Cycle Trigger source by all the Camera and Illumination Controllers.
- As Cycle Sequence Trigger source by all the Camera and Illumination Controllers.
- As event source by all the Divider tools of the I/O Toolbox.
- As event source by all the Multiplier/Divider tools of the I/O Toolbox
- As event source for both channels of all the Delay) tools of the I/O Toolbox.

User output register

Coaxlink products provide a User Output Register where bits are named `UserOutput0` … `UserOutput7`.

> **NOTE**
> **1630 Coaxlink Mono** implements only the 4 lowest bits!

Any User Output Register bit can be used asa signal source by any output-capable GPIO line.

The user application has two options to define the state of the User Output Register bits:

- The User Actions option allows to change the state of each bit individually.
- Setting a value to the UserOutputValueAll to define the state of all bits.

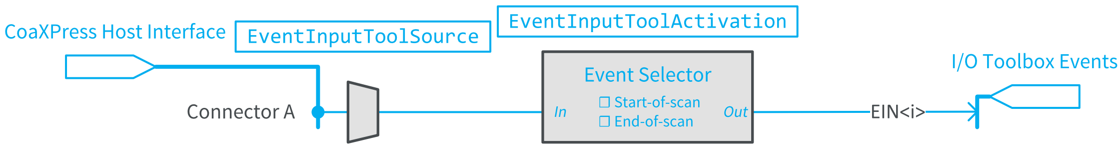
Getting the value of UserOutputValueAll allows the user application to get the state of all bits.

9.9. Event Input Tool

Applies to:
QuadG3DF

Tool Name	Short Name	Inputs Count/Type	Outputs Count/Type/Name
Event Input Tool	EIN	1 Event	1 event stream: EIN<i>

Diagram



EIN tool functional and wiring diagram

Connector A of the CoaXPress Host Interface is the only source of events that can be selected.

The tool feeds one I/O Toolbox event stream named EIN<i>.

Operation

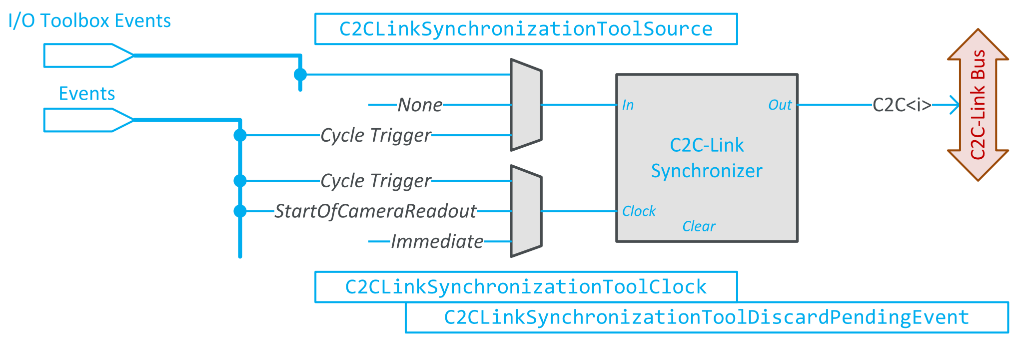
The Event Input tool decodes the custom CoaXPress GPIO message received on the CoaXPress connector A of a slave **1635 Coaxlink Quad G3 DF**.

The Event Input tool delivers one event on the reception of a "Start-of-scan message" or an "End-of-scan message" according to the EventInputToolActivation settings.

9.10. C2C-Link Synchronization Tool

Tool Name	Short Name	Inputs Count/Type	Outputs Count/Type/Name
C2C-Link SynchronizerTool	C2C	1 or 2 event streams	1 event stream: C2C<i>

Diagram



C2C tool functional and wiring diagram

The C2C-Link Synchronizer tool (C2C) tool delivers one event stream to the C2C-Link Bus driver. It includes the following blocks:

- A source selector
- A clock source selector
- An event synchronizer with clear control

Operation

Source selector

The source selector selects the event stream applied to the tool input (In). It provides following options:

- On C2C1 instance only: Cycle Trigger event stream driven by the Camera and Illumination controller.
- On C2C2 and C2C3 instances only: any I/O toolbox event.

Synchronizer control

The clock source selector controls the event stream synchronization:

- When C2CLinkSynchronizationToolClock is set to Immediate, the event stream applied to the input (In) is sent immediately to the output.

- On C2C2 and C2C3 instances only: when C2CLinkSynchronizationToolClock is set to CycleTrigger, the event is latched and delayed until the following Cycle Trigger event.
- On C2C2 and C2C3 instances only: when C2CLinkSynchronizationToolClock is set to StartOfCameraReadout, the event is latched and delayed until the following following Start of Camera Readout event.

The C2CLinkSynchronizationToolDiscardPendingEvent command discards an event that has been received but that has not been forwarded.

NOTE
Area-scan firmware variants provide 2 instances of the C2C tool; line-scan firmware variants provide 3 instances!

10. Event Signaling And Counting

10.1. Introduction

Short description

Coaxlink products feature a powerful event management that allows the application to be notified of the occurrence of various events.

In addition to the GenTL standard `EVENT_NEW_BUFFER` event, the Coaxlink GenTL producer provide a wide set of custom event sources.

The event sources are grouped by types according to the function block and the GenTL module they belong to.

Each custom event source is associated with a counter that counts the number of occurrences.

For each notified custom event, the following event context data is recorded and made available to the application:

- Identifier of the event source
- Time stamp (expressed in microseconds)
- 3 user-defined context data

Each individual event source is configurable:

- The event notification can enabled or disabled.
- The content of each user-defined context data.

Event data are temporarily stored in the Event Queue Buffer. The Coaxlink driver is notified, using an interruption mechanism, of the availability of one or more event entries in the Coaxlink Event Queue Buffer.

The Coaxlink driver implements the GenTL signaling mechanism for reporting the occurrence of asynchronous events to the application software.

The EGrabber API provides 3 callback threading models:

- *CallbackOnDemand:* This is the simplest model which gives complete control over when and how callbacks are invoked. Events are processed on demand.
- *CallbackSingleThread*: This model delivers events to callbacks in their chronological order, sequentially, in a dedicated thread context. Events are processed automatically as soon as they are available.
- *CallbackMultiThread*: This model delivers events to callbacks in separate threads (one thread per event DATA type). Events are processed automatically as soon as they are available.

Event types

GenTL identifies the events according to their Type and their relevant object Module.

Standard event types

The Coaxlink driver implements the following standard event type for registration by the GenTL Consumer application:

GenTL Standard Event Type	GenTL Module	Description
EVENT_NEW_BUFFER	Data Stream	Notification on newly filled buffers.

Custom event types

Beside the *standard event types*, the GenTL specification provides room for *custom event types*.

Custom event types are specific to the GenTL Producer implementation (Coaxlink driver in this case).

The Coaxlink driver implements the following custom event types for registration by the GenTL Consumer application:

Event Type	Module	Description
EVENT_CUSTOM_IO_TOOLBOX	Interface (Device)	Notification of I/O Toolbox events
EVENT_CUSTOM_CXP_INTERFACE	Interface (Device)	Notification of CoaXPress Host Interface events
EVENT_CUSTOM_CIC	Device	Notification of Camera and Illumination Control events
EVENT_CUSTOM_DATASTREAM	Data Stream	Notification of CoaXPress data stream events

NOTE
The EVENT_CUSTOM_IO_TOOLBOX and the EVENT_CUSTOM_CXP_INTERFACE event types can also be registered on a Device Module.

NOTE
The custom event types are generic; each one gathers multiple event sources.

See also: "Custom Events Sources" on page 161 for an exhaustive list.

Custom events counter

A 32-bit counter is associated with every custom event source.

The counter cannot be disabled. When it reaches its maximum value, 4 294 967 295 (2^{32} - 1), it wraps around to 0.

At any time, the user application can:

- Read the count value of a selected event source.
- Reset the counter of a selected event source.
- Reset the counters of all the event sources of the module.

The count-value can also be used as user-defined context data by any event source.

Custom events configuration

The event source is configurable.

At any time, the user application can:

- Enable or disable the notification of a selected event source.
- Enable or disable the notification of all the event sources of the module.
- Define the content of each user-defined context data of a selected event source..

Notification

By default, all notifications are disabled.

The application software must configure the event notification filter according the application needs.

The configuration of the notification filter configuration can be modified at any time without interfering with the event counting function.

Context data

The last 3 32-bit context data words of the event context data can be configured as follows:

- Event-specific data.
- State of I/O lines sampled at the event occurrence time
- Count value of any event counter.
- Count value of any Quadrature Decoder (QDC) position counter.

Some event sources provide additional options.

See also: "Event Specific Context Data" on page 165

10.2. Custom Events Sources

Data Stream Event Sources (Data Stream Module)

EVENT_CUSTOM_DATASTREAM

Event Source	Description
DATASTREAM_START_OF_CAMERA_READOUT	The first pixel data of an image frame is written into the on-board FIFO Buffer. Applies to area-scan firmware variants only.
DATASTREAM_END_OF_CAMERA_READOUT	The last pixel data of an image frame is written into the on-board FIFO Buffer. Applies to area-scan firmware variants only.
DATASTREAM_START_OF_SCAN	The first pixel data of an image scan is written into the on-board FIFO Buffer. Applies to line-scan firmware variants only.
DATASTREAM_END_OF_SCAN	The last pixel data of an image scan is written into the on-board FIFO Buffer. Applies to line-scan firmware variants only.
DATASTREAM_REJECTED_FRAME	An image frame is rejected (On-board FIFO Buffer is full) Applies to area-scan firmware variants only.
DATASTREAM_REJECTED_SCAN	An image scan is rejected (On-board FIFO Buffer is full) Applies to line-scan firmware variants only.

I/O Toolbox custom events (Interface module)

EVENT_CUSTOM_IO_TOOLBOX event sources

Event Source	Description
IO_TOOLBOX_LIN1	Line Input Tool 1 – Event output
IO_TOOLBOX_LIN2	Line Input Tool 2 – Event output
IO_TOOLBOX_LIN3	Line Input Tool 3 – Event output
IO_TOOLBOX_LIN4	Line Input Tool 4 – Event output
IO_TOOLBOX_LIN5	Line Input Tool 5 – Event output
IO_TOOLBOX_LIN6	Line Input Tool 6 – Event output
IO_TOOLBOX_LIN7	Line Input Tool 7 – Event output
IO_TOOLBOX_LIN8	Line Input Tool 8 – Event output
IO_TOOLBOX_QDC1	Quadrature Decoder Tool 1 – Event output
IO_TOOLBOX_QDC1_DIR	Quadrature Decoder Tool 1 – Changed direction
IO_TOOLBOX_QDC2	Quadrature Decoder Tool 2 – Event output
IO_TOOLBOX_QDC2_DIR	Quadrature Decoder Tool 2 – Changed Direction
IO_TOOLBOX_DIV1	Divider Tool 1 – Event output
IO_TOOLBOX_DIV2	Divider Tool 2 – Event output
IO_TOOLBOX_MDV1	Multiplier/Divider Tool 1 – Event output
IO_TOOLBOX_MDV2	Multiplier/Divider Tool 2 – Event output
IO_TOOLBOX_DEL11	Delay Tool 1 Output 1 – Event output
IO_TOOLBOX_DEL12	Delay Tool 1 Output 2 – Event output
IO_TOOLBOX_DEL21	Delay Tool 2 Output 1 – Event output
IO_TOOLBOX_DEL22	Delay Tool 2 Output 2 – Event output
IO_TOOLBOX_USER_EVENT_1	User Event 1
IO_TOOLBOX_USER_EVENT_2	User Event 2
IO_TOOLBOX_USER_EVENT_3	User Event 3
IO_TOOLBOX_USER_EVENT_4	User Event 4
IO_TOOLBOX_EIN1	Event Input Tool 1 – Event output
IO_TOOLBOX_EIN2	Event Input Tool 2 – Event output

NOTE
Check the "I/O Toolbox Composition" on page 137 for applicable values

CoaXPress Host Interface Custom Events (Interface module)

EVENT_CUSTOM_CXP_INTERFACE

Event Source	Description
CXP_INTERFACE_CRC_ERROR_CXP_A	A CRC error is detected on the Connection A of the CoaXPress Host Interface.
CXP_INTERFACE_CRC_ERROR_CXP_B	A CRC error is detected on the Connection B of the CoaXPress Host Interface.
CXP_INTERFACE_CRC_ERROR_CXP_C	A CRC error is detected on the Connection C of the CoaXPress Host Interface.
CXP_INTERFACE_CRC_ERROR_CXP_D	A CRC error is detected on the Connection A of the CoaXPress Host Interface.

Camera and Illumination Controller Event Sources (Device)

EVENT_CUSTOM_CIC event sources

Event Source	Description
CIC_CAMERA_TRIGGER_RISING_EDGE	Rising edge of the Camera Trigger output signal (Start of Exposure in RC and RG camera control methods)
CIC_CAMERA_TRIGGER_FALLING_EDGE	Falling edge of the Camera Trigger output signal (End of Exposure in RG camera control method)
CIC_STROBE_RISING_EDGE	Rising edge of the Strobe output signal
CIC_STROBE_FALLING_EDGE	Falling edge of the Strobe output signal
CIC_ALLOW_NEXT_CYCLE	A new camera cycle is allowed to start immediately.
CIC_DISCARDED_CIC_TRIGGER	A CIC cycle trigger is discarded.
CIC_PENDING_CIC_TRIGGER	A CIC cycle trigger is recorded, but its execution is delayed until CIC is ready.
CIC_CXP_TRIGGER_ACK	A positive acknowledgment is received in response to a CoaXPress Host to Device Trigger Message.
CIC_CXP_TRIGGER_RESEND	A resent of the CoaXPress Host to Device Trigger Message is executed.
CIC_TRIGGER	CIC trigger

NOTE
There is one Camera and Illumination Controller instance per GenTL Device Module. The number of GenTL Device Modules per Coaxlink card is defined by the firmware-variant.

10.3. Event Specific Context Data

EVENT_DATA_NUMID_CIC_DISCARDED_CIC_TRIGGER

Value of EventSpecific for EVENT_DATA_NUMID_CIC_DISCARDED_CIC_TRIGGER is a bitfield that can be interpreted according to the following definitions:

Bit#	Description
0	Cause: image buffer is full.
1	Cause: camera cycle not complete.
2	Cause: maximum number of pending triggers already reached.
3	Cause: data stream is not active

EVENT_DATA_NUMID_CIC_PENDING_CIC_TRIGGER

Value of EventSpecific for EVENT_DATA_NUMID_CIC_PENDING_CIC_TRIGGER is a bitfield that can be interpreted according to the following definitions:

Bit#	Description
0	Cause: image buffer is full.
1	Cause: camera cycle not complete

EVENT_DATA_NUMID_DATASTREAM_START_OF_SCAN

Value of EventSpecific for EVENT_DATA_NUMID_DATASTREAM_START_OF_SCAN is a bitfield that can be interpreted according to the following definitions:

Bit#	Description
1	Cause: software trigger.
2	Cause: hardware trigger
3	Cause: `DSStartAcquisition` or end of previous scan

EVENT_DATA_NUMID_DATASTREAM_END_OF_SCAN

Value of EventSpecific for EVENT_DATA_NUMID_DATASTREAM_END_OF_SCAN is a bitfield that can be interpreted according to the following definitions:

Bit#	Description
1	Cause: software trigger.
2	Cause: hardware trigger
3	Cause: Cause: reached scan length
4	Cause: `DSStopAcquisition`
5	Cause: internal exception (image buffer almost full)

EVENT_DATA_NUMID_DATASTREAM_REJECTED_FRAME

Value of EventSpecific for EVENT_DATA_NUMID_DATASTREAM_REJECTED_FRAME is a bitfield that can be interpreted according to the following definitions:

Bit#	Description
0	Cause: image buffer is full.
1	Cause: data stream is not active

10.4. About GenTL Signaling

The Coaxlink driver implements the Signaling mechanism of a GenTL Provider.

This mechanism is briefly described hereafter.

See also: section 4.2 starting on page 34 of the GenICam GenTL Standard Version 1.4 for an extensive description.

Event Registration

Source: GenTL specification

Before the GenTL Consumer can be informed about an event, the event object must be registered. After a module instance has been created in the enumeration process an event object can be created with the `GCRegisterEvent()` function. This function returns a unique EVENT_HANDLE which identifies the registered event object. To get information about a registered event the `EventGetInfo()` function can be used.

WARNING
There must be only one event registered per module and event type!

(…)

After an EVENT_HANDLE is obtained the GenTL Consumer can wait for the event object to be signaled by calling the `EventGetData()` function. Upon delivery of an event, the event object carries data. This data is copied into a GenTL Consumer provided buffer when the call to `EventGetData()` was successful.

Notification and Data Retrieval

Source: GenTL specification

If the event object is signaled, data was put into the event data queue at some point in time. The `EventGetData()` function can be called to retrieve the actual data.

(…)

When data is read with this function the data is removed from the queue. Afterwards the GenTL Producer implementation checks whether the event data queue is empty or not. If there is more data available the event object stays signaled and next the call to `EventGetData()` will deliver the next queue entry. Otherwise the event object is reset to not signaled state.

(…)

The exact type of data is dependent on the event type and the GenTL Producer implementation. The data is copied into a user buffer allocated by the GenTL Consumer. The content of the event data can be queried with the `EventGetDataInfo()` function. The maximum size of the buffer to be filled is defined by the event type and can be queried using EVENT_INFO_DATA_SIZE_MAX after the buffer is delivered. This information can be queried using the `EventGetInfo()` function.

11. Advanced Features

11.1. Sub-link Acquisition

Applies to:
QuadG3 QuadG3LH

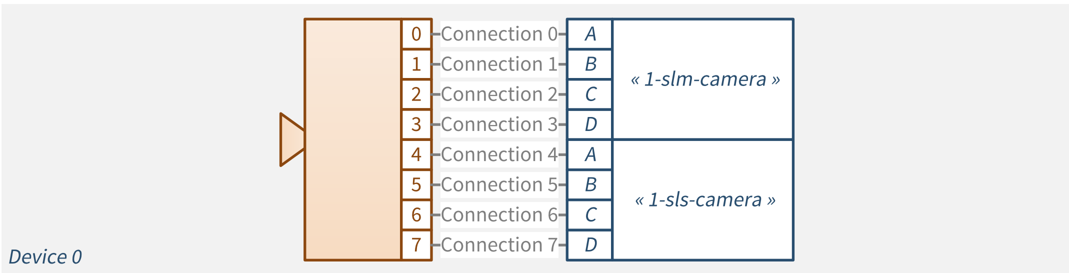
WARNING
This function is only available for a restricted set of product and firmware variants combinations!

See also: "Firmware Variants per Product" on page 20 and search for *"sls-camera"* and *"slm-camera"* in the Firmware Variant column.

Principles

The *Sub-link Acquisition* feature allows to acquire images from specific 8-connection CoaXPress cameras.

The 8-connection CoaXPress link is divided into 2 sub-links. Each sub-link connects to a Coaxlink card using the 1D8SL4 connection scheme:



8-connection camera using 2 sub-links and 2 Coaxlink cards

The first 4-connection sub-link connects to the "sub-link-master grabber": a Coaxlink card fitted with the *1-slm-camera* firmware variant.

The next 4-connection sub-link connects to the "sub-link-slave grabber": a Coaxlink card fitted with the *1-sls-camera* firmware variant.

Each grabber delivers one-half of the image frame into a GenTL buffer. The application has to reconstruct the whole image frame by merging the contents of the two corresponding buffers. This is shown in the `320-sublink` EGrabber sample program.

The master grabber controls the camera and manages the system triggers.

Both grabbers are configured to capture all the image data of their respective sub-link.

Camera requirements

This feature applies only to 8-connection area-scan cameras having the following characteristics:

- The image header and the image data of the first line are packed together into a single CoaXPress data packet and delivered to CoaXPress Connection 0.
- For the remaining lines of the frame (or ROI), the image data of a single image line are packed into a single packet and delivered to the next CoaXPress Connection. Connections are rotated using the CoaXPress standard packet distribution ordering rule: (0 to 7, then back 0).
- The image frame (or ROI) height must be a multiple of 8 lines to ensure that the last image line is delivered on the last connection (Connection 7).

11.2. Multi-Stream Acquisition

Applies to:
QuadG3 QuadG3LH

WARNING
This function is only available for a restricted set of product and firmware variants combinations!

See also: "Firmware Variants per Product" on page 20 and search for *"-data-stream" suffix* in the Firmware Variant column.

4-data-stream Concurrent Acquisition

The *1-camera, 4-data-stream* firmware variant of **1633 Coaxlink Quad G3** and **1633-LH Coaxlink Quad G3 LH** allows to connect one area-scan CoaXPress camera that delivers up to 4 independent data streams.

The frame grabber sorts the incoming CoaXPress data blocks according to the value of the 2 least significant bits of the CoaXPress `StreamID` and feeds four independent data paths.

Each data path is capable of handling the full CoaXPress link bandwidth, namely 2.5 Gigabytes/s.

It includes:

1. A 250 MB FIFO buffer memory
2. A pixel processor that allows to align 10/12/14-bit data to the LSB or to the MSB of a 16-bit container.
3. A DMA engine that transfers image data directly to the user memory space of the Host PC.

The `multistream` sample program shows how to create 4 instances of EGrabber and start acquisition on 4 concurrent data streams.

11.3. CoaXPress Data Forwarding

Applies to:

QuadG3DF

WARNING

This function is only available for a restricted set of product and firmware variants combinations!

See also: "Firmware Variants per Product" on page 20 and search for *"-df-"* in the Firmware Variant column.

Data Forwarding Principles

The data forwarding capabilities of **1635 Coaxlink Quad G3 DF** allows to forward the image data from a camera to multiple frame grabbers in different Host PC's.

Data forwarding

A *DF-capable* card, such as the **1635 Coaxlink Quad G3 DF**, forwards the data received on the CoaXPress Host connector to the CoaXPress Data Forwarding connector.

The image data packets embedded in the serial bit stream on connections A, B, C, D of the CoaXPress Host connector are forwarded to the connections FA, FB, FC, FD of the CoaXPress Data Forwarding connector.

The serial bit streams are re-timed to operate always at CXP-6 speed regardless the link speed of the camera. Therefore, idle characters are, when necessary, removed or added in the bit stream. Addition or removal of idle characters doesn't affect the payload. Image data are preserved, including CRC's. For proper operation of data forwarding, it is mandatory that the camera inserts one IDLE word at least once every 100 words as required by the Coaxlink 1.1 standard §8.2.5.1.

The image data are retransmitted with a negligible latency: typically, a few periods of the 32-bit character transmission time.

NOTE
The Data Forwarding output port doesn't comply with the specification of a CoaXPress Device! It can only feed another **1635 Coaxlink Quad G3 DF**.

Data Forwarding Chain

A *DF-chain* is composed of 2 or more data-forwarding capable cards where the CoaXPress Data Forwarding connector of one card is connected to the CoaXPress Host connector of the next card using a set of 1, 2 or 4 coaxial cables named *DF-bridge*.

NOTE
There are no specified upper limit to the number of cards in a DF-chain.

The camera is attached to the CoaXPress Host connector of the first card of the DF-chain, this card is named **DF-master**. The other cards, of the DF-chain are named **DF-slaves**. The CoaXPress Data Forwarding connector of the last DF-slave card is left unconnected.

CoaXPress Link discovery and configuration

The DF-master card is responsible for the discovery and the configuration of the CoaXPress Link of the camera.

The CoaXPress Host Interface of the DF-slaves are automatically configured with the same number of connections as discovered by the DF-Master.

For instance if the camera uses two connections, only two connections are required for every DF-bridge.

Firmware variants

The firmware variant to install on the DF-master must be selected according to the camera type:

- For an area-scan camera, install the `1-camera` firmware variant.
- For a line-scan camera, install the `1-camera, line-scan` firmware variant.

The firmware variant to install on the DF-slaves must match the firmware variant installed on the DF-master:

- When the `1-camera` firmware variant is installed on the DF-master, install the `1-df-camera` firmware variant on all DF-slaves.
- When `1-camera, line-scan` firmware variant is installed on the DF-master, install the `1-df-camera, line-scan` firmware variant on all DF-slaves.

Data Acquisition Control

For correct operation of data forwarding, the application must respect the following rules:

- The data acquisition must be activated on all the DF-slaves before being activated on the DF-master.
- The data acquisition must be de-activated on the DF-master before being de-activated on the DF-slaves.

For line-scan applications only, a start-of-scan and end-of-scan synchronization mechanism is implemented to ensure that all the cards of the DF-chain can capture the same lines of image data.

See also: "Line-scan Triggers Synchronization" on page 177

Camera Cycle Control

If required by the application, the DF-master card is responsible for the elaboration of the CoaXPress Host-to-Device trigger. This is achieved in the same way as for non-data-forwarding Coaxlink cards.

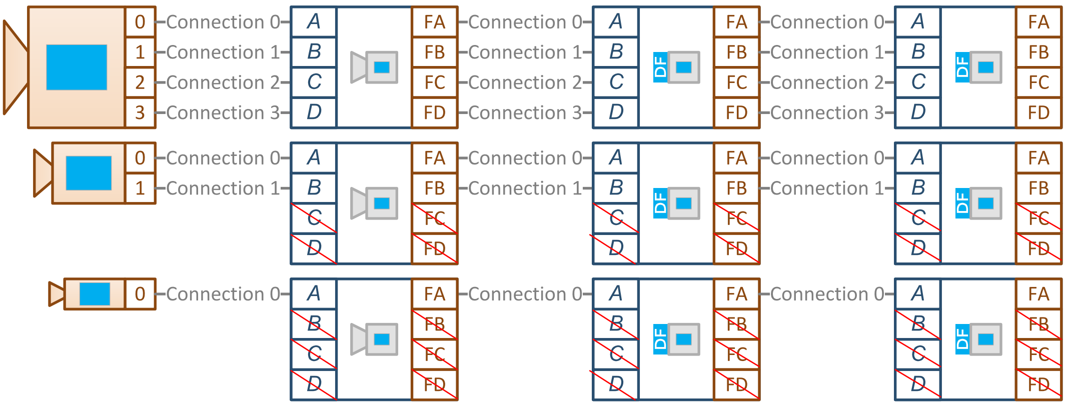
See also: "CoaXPress Host To Device Trigger" on page 39

Data Forwarding Connection Schemes

Area-scan Camera Data Forwarding

The following drawing illustrates 3 connection schemes where the image data of an area-scan camera is forwarded to 3 Host PCs: one for a 4-connection camera, one for a 2-connection camera, one for a single-connection camera.

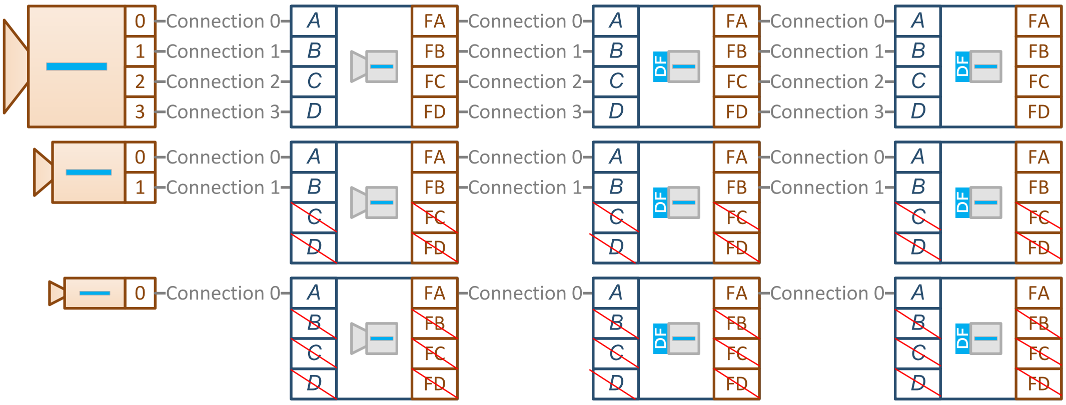
The first **1635 Coaxlink Quad G3 DF** card must be configured with a `1-camera` firmware variant; the other must be configured with the `1-df-camera` firmware variant.



Line-scan Camera Data Forwarding

The following drawing illustrates 3 connection schemes where the image data of a line-scan is forwarded to 3 Host PCs: one for a 4-connection camera, one for a 2-connection camera, one for a single-connection camera.

The first **1635 Coaxlink Quad G3 DF** card must be configured with a `1-camera, line-scan` firmware variant; the other must be configured with the `1-df-camera, line-scan` firmware variant.



Line-scan Triggers Synchronization

The start-of-scan and end-of-scan trigger synchronization mechanism ensures that all the cards of the DF-chain capture the same lines of image data.

Data Forwarding - Master Card

In a DF-chain, the DF-master card forwards its start-of-scan and end-of-scan events to the DF-slave cards.

The generation of the start-of-scan and end-of-scan events on the DF-master is achieved in the same way as for non-data-forwarding Coaxlink cards.

See also: "Line Scan Acquisition" on page 236

The DF-master card:

- First, synchronizes the asynchronous scan triggers on the next start-of line image data.
- Then, share the synchronized scan triggers with all the DF-slaves.

The sharing of the scan triggers is achieved by the insertion of high-priority "custom GPIO messages" in the bit stream. These messages are forwarded by all the DF-slaves together with the image data.

Data Forwarding - Slave Card(s)

On reception of such a message, the DF-slave generates a hardware event. Two kind of events are possible:

- Start-of-scan event.
- End-of-scan event

These events are available through the "Event Input Tool" on page 154 of the I/O Toolbox.

For applications requiring synchronized line-scan acquisition, the I/O toolbox EIN tools of the DF-slaves must be used as local start-of-scan and end-of-scan trigger sources.

Configuration Script Example

The following script configures Data Forwarding frame grabbers for synchronized line-scan acquisition:

```
for (var grabber of grabbers) {
    if (grabber.InterfacePort.get("InterfaceID").includes('df-camera')) {
        console.log("Configuring slave card");
        // set the Width/Height/PixelFormat of the (virtual) remote device (on
        // the slave card) equal to the Width/Height/PixelFormat of the (real)
        // camera (connected to the master card)
        grabber.RemotePort.set("Width", 8192);
        grabber.RemotePort.set("Height", 1);
        grabber.RemotePort.set("PixelFormat", "Mono8");
        // configure the event input tool EIN1
        grabber.InterfacePort.set("EventInputToolSource[EIN1]", "A");
        grabber.InterfacePort.set("EventInputToolActivation[EIN1]", "StartOfScan");
        // configure the event input tool EIN2
        grabber.InterfacePort.set("EventInputToolSource[EIN2]", "A");
        grabber.InterfacePort.set("EventInputToolActivation[EIN2]", "EndOfScan");
        // configure start/end of scan triggers
        grabber.StreamPort.set("StartOfScanTriggerSource", "EIN1");
        grabber.StreamPort.set("EndOfScanTriggerSource", "ScanLength");
        grabber.StreamPort.set("ScanLength", 1000);
    } else {
        console.log("Configuring master card");
        grabber.RemotePort.set("TestPattern", "GreyDiagonalRampMoving");
        grabber.RemotePort.set("CxpLinkConfiguration", "CXP6_X4");
        grabber.RemotePort.set("CxpLinkConfigurationPreferredSwitch", "CXP6_X4");
        grabber.RemotePort.set("TriggerSource", "CXPin");
        grabber.RemotePort.set("TriggerMode", "On");
        grabber.DevicePort.set("CameraControlMethod", "RG");
        grabber.DevicePort.set("ExposureReadoutOverlap", "True");
        grabber.DevicePort.set("CxpTriggerAckTimeout", "0");
        grabber.DevicePort.set("StrobeDuration", "0");
        grabber.DevicePort.set("ExposureTime", "20");
        grabber.DevicePort.set("ExposureRecoveryTime", "0");
        grabber.DevicePort.set("CycleMinimumPeriod", "50");
        // configure start/end of scan triggers
        grabber.StreamPort.set("StartOfScanTriggerSource", "Immediate");
        grabber.StreamPort.set("EndOfScanTriggerSource", "ScanLength");
        grabber.StreamPort.set("ScanLength", 1000);
    }
}
```

NOTE
In this example, the start-of-scan trigger is the receipt of the start-of-scan event from the master, but the end-of-scan trigger is generated locally. One alternative would be to use `EIN2` as `EndOfScanTriggerSource`.

11.4. Flat Field Correction

Applies to:

QuadG3	QuadG3LH	QuadG3DF	Octo

WARNING

This function is only available for a restricted set of product and firmware variants combinations!

See also: "Firmware Variants per Product" on page 20 and search for *FFC* in the *Advanced Processing* column.

What is Flat Field Correction?

The *Flat-field correction* (Wikipedia: FFC) is a method used to correct:

- the differences of light sensitivity between the pixel sensors of a camera
- some artifacts related to the optical system (e.g., non-uniform lighting and vignetting)

The goal is to correct the pixels of the captured (raw) images in such a way that when a uniform background is captured by the system (camera & lens), the resulting output image is uniform.

Formula

This correction is achieved by applying the following operation to each pixel of the raw image:

$$CorrectedPixel = (RawPixel - Offset) * Gain$$

where both *Offset* and *Gain* coefficients are specific values for each pixel.

The evaluation of the coefficients *Offset* and *Gain* requires a calibration procedure explained in the next paragraph.

Calibration

The calibration procedure to compute the coefficients is done in two steps:

Dark image acquisition

A dark image is acquired by the system. This is typically achieved by covering the lens with the cap. The captured image represents the dark current of the sensors, and is considered as a fixed bias that we want to eliminate when acquiring images in normal conditions. This correction is called dark-frame subtraction.

$$\text{CorrectedPixel} = \text{RawPixel} - \text{DarkPixel}$$

For each pixel, the *DarkPixel* value corresponds to the *Offset* in the above FFC.

Flat image acquisition

A flat image is acquired by the system. For example by capturing a flat (uniform) background, not too bright to avoid saturation and not too dark.

From dark and flat acquisitions, we have enough data to compute the *Gain* value of the FFC.

For this we define *CorrectedPixel* as the pixel value we consider to be correct for the flat image. Let's set this value as the average pixel value of the flat image (*average(Flat)*), corrected by the average of the dark image (*average(Flat)*). In the FFC terms, this gives:

$$\text{average(Flat)} - \text{average(Dark)} = (\text{FlatPixel} - \text{DarkPixel}) * \text{Gain}$$

This leads to the Gain value

$$\text{Gain} = (\text{average(Flat)} - \text{average(Dark)}) / (\text{FlatPixel} - \text{DarkPixel})$$

The same computation is repeated (*width * height*) times, to cover all pixels of both flat and dark images. This results in a specific correction for each pixel of the image.

NOTE
Note: this calibration procedure must be redone if any part of the system is changed, including the camera unit, lighting or optics equipment.

Calibration of color pixel formats

For color pixel formats, we have several ways of computing the value of *average(Flat)*. In all cases, the *Gain* computation is repeated (*width * height * componentsPerPixel*) times to cover all pixel components, which results in specific corrections for each pixel component of the image.

Handling pixel components individually

I.e. in `RGB`:

- using *average(Flat[Red])* for computing the *Gain* values of *Red* components;
- using *average(Flat[Green])* for computing the *Gain* values of *Green* components;
- using *average(Flat[Blue])* for computing the *Gain* values of *Blue* components.

Handling pixel components together

I.e. in *RGB*, using *average(average(Flat[Red]), average(Flat[Green]), average(Flat[Blue]))* for computing the *Gain* values of *Red*, *Green* and *Blue* components.

This way of computing the average (i.e. over the pixel components) results in FFC coefficients that also correct the balance between components. Therefore, depending on the quality of the uniform background used to acquire the flat image, the FFC can effectively perform a white balance correction.

Coaxlink FFC

Implementation of Flat Field Correction in Coaxlink products

Some cameras have a built-in FFC module while other cameras do not implement this feature, The devices without that functionality can however be corrected by the FFC core of the Coaxlink card.

The FFC core of the Coaxlink firmware corrects the pixels directly coming from the camera by applying the FFC using the coefficients (*Offset* and *Gain*) corresponding to their locations in the image. Because the correction happens at a very early stage in the Coaxlink pixel processing chain, the other pixel processing functions such as *RedBlueSwap*, *LUT*, and *Bayer Decoding* are performed on corrected pixels.

Gain and Offset Coefficients Format

The coefficients calculated in the calibration procedure can be loaded into the Coaxlink card, provided they are encoded as follows:

- *Offset* and *Gain* values for one pixel component are packed together into a 16-bit little-endian value:
  - *Gain* is encoded in Wikipedia:UQ2.8 on bits 9..0
  - *Offset* is a 6-bit unsigned integer on bits 15..10
- Coefficients related to pixel component values are treated separately in the same sequence as the pixel components of the image. For example in *RGB8* format, one pixel is encoded as 3 successive 8-bit values (*Red*, *Green*, *Blue*), therefore we need 3 successive 16-bit packed coefficients to correct one `RGB8` pixel.

If the 16-bit packed coefficients are stored (in sequence) in a binary file (let's say '*path/to/coefficients.ffc*'), they can be easily loaded from a Euresys script by calling

```
require("coaxlink://ffc/load")(grabber, 'path/to/coefficients.ffc');
```

where *grabber* is the script variable referencing the grabber to configure.

NOTE
such a binary file can be created by the Euresys `ffc-wizard` sample application

Specifications

Camera Types

The FFC feature is applicable to monochrome, Bayer CFA and RGB Color area-scan cameras delivering 8-, 10-, 12- 14- or 16-bit data per pixel component.

Maximum Image Size

The following table shows the maximum image size for all supported pixel formats when FFC is enabled :

Bit depth	Maximum Image size [Pixels]		
	Monochrome and Bayer CFA formats	RGB formats	RGBa formats
8-bit	134,217,728 pixels	44,739,242 pixels	33,554,432 pixels
10-bit	107,374,182 pixels	35,791,394 pixels	26,843,545 pixels
12-bit	89,478,485 pixels	29,826,161 pixels	22,369,621 pixels
14-bit	76,695,844 pixels	25,565,281 pixels	19,173,961 pixels
16-bit	67,108,864 pixels	22,369,621 pixels	16,777,216 pixels

Performance

When enabled, the FFC feature adds a significant load to the DRAM memory since it fetches an additional 16-bit coefficient data for each processed pixel. When the FFC is enabled, the Coaxlink cards are only able to sustain a fraction of the maximum data rate achievable by the CoaXPress Link. This dimension-less value is named "*Sustainable relative data rate*"

The following table shows the *sustainable relative data rate* for all bit depths and product/firmware variant combinations supporting FFC.

Sustainable Relative Data Rate

Bit depth	Sustainable Relative Data Rate [% of a 4-lane CXP-6 maximum data rate]	
	1633 Coaxlink Quad G3 - 1-camera 1633-LH Coaxlink Quad G3 LH - 1-camera	3602 Coaxlink Octo - 2-camera
8-bit	70.0 %	123.2 %
10-bit	77.8 %	136.9 %
12-bit	84.0 %	147.8 %
14-bit	89.1 %	156.8 %
16-bit	93.3 %	164.3 %

NOTE
The "Sustainable Relative Data Rate" is global for all cameras attached to a board.For instance, in the **3602 Coaxlink Octo** - 2-camera 8-bit use case, the sustainable data rate of 123.2% can be split unequally to 100% for a camera and 23.2% for the other camera.

NOTE
Coaxlink cards do not acquire any data during blanking intervals. Line- and frame-blanking intervals do not consume memory bandwidth and therefore must be excluded in the calculation of the camera data rate.

To avoid latencies, FIFO buffer overflow and loss of frames, Euresys recommends to limit the (global) camera data rate accordingly.

Enabling the FFC

In the Data Stream module, set the FfcControl feature value to Enable.

Disabling the FFC

In the Data Stream module, set the FfcControl feature value to Disable.

FFC Wizard Sample Program

Euresys provides the source code of a sample application, called `ffc-wizard`, that computes the coefficients and packs them in a binary file targeting the Coaxlink cards.

The purpose of this sample code is threefold:

- guide the user through the calibration procedure;
- provide a technical and practical translation of what's described in this document;
- provide building blocks for developing custom applications.

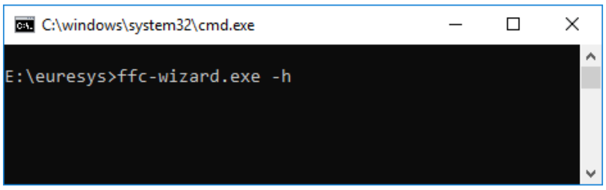
Building

The source code lies in a single source file: `src/ffc-wizard.cpp`. Building the application should be straightforward;

- for Windows, there is a Microsoft Visual Studio project file `ffc-wizard.vcproj`;
- for Linux and macOS, a `Makefile` is provided.

Usage

The wizard is a console application. The help message is displayed when the flag `-h` (or `--help`) is given:



getting the help message

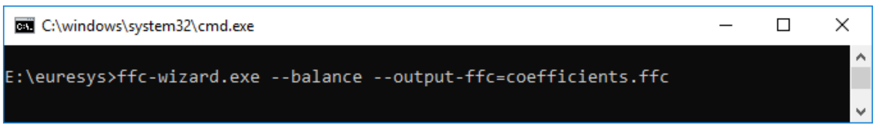
```
> ffc-wizard.exe --help
Flat Field Correction Wizard
fcc-wizard [OPTIONS]

Options:
  --if=INT                 Index of GenTL interface to use
  --dev=INT                Index of GenTL device to use
  --ds=INT                 Index of GenTL data stream to use
  --average=INT            Number of images to average (default: 10)
  --roi_x=INT              Horizontal offset in pixels of ROI upper-left corner
(default: 0)
  --roi_y=INT              Vertical offset in pixels of ROI upper-left corner
(default: 0; ignored for line-scan)
  --roi_width=INT          Width of ROI (default: whole image)
  --roi_height=INT         Height of ROI (default: whole image; ignored for line-scan)
  --balance                Compute flat image average on all components rather than on
each component
  --linescan               Force line-scan mode i.e. average image lines
(automatically enabled for line-scan cards)
  --dark-setup=SCRIPT      Path to setup script for dark acquisitions
  --flat-setup=SCRIPT      Path to setup script for flat acquisitions
  --timeout=MS             Maximum time in milliseconds to wait for an image (default:
1000)
  --dark-histogram=FILE    Path to histogram html page of average dark image to output
and open
  --flat-histogram=FILE    Path to histogram html page of average flat image to output
and open
  --output-ffc=FILE        Path to coefficients output file (Coaxlink ffc binary
format)
  --load-ffc=FILE          Load coefficients into Coaxlink coefficients partition
(default: computed coefficients)
  --no-interact            Skip user interaction
  -h  --help               Display help message

Note: the ROI options allow defining a rectangular region to consider while computing
averages,
      this is useful to eliminate pixels close to borders in images subject to
vignetting.
```

Options details

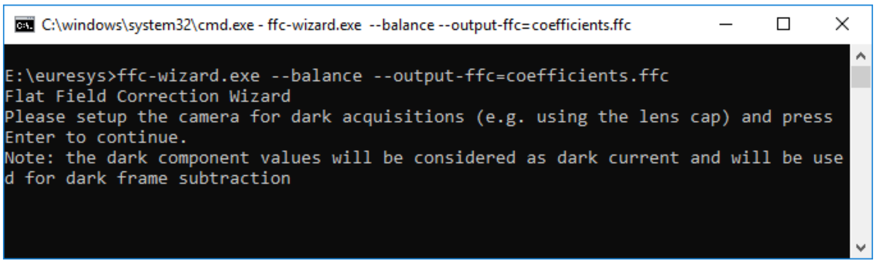
Flags	Details
`--if`, `--def`, `--ds`	the indexes of the GenTL modules that identify the data stream to use (and/or to configure)
`--average`	the number of images to acquire for dark and flat acquisitions; only the average of those acquisitions is further used in the calibration procedure
`--roi_x`, `--roi_y`, `--roi_width`, `--roi_height`	an optional rectangular region to consider when computing the average pixel value of the flat image (*average(Flat)*); this impacts the evaluation of the gain value for each pixel
`--balance`	enables the white balance; i.e. whether the coefficients of color pixel components are computed to balance the component values or not; obviously, this requires the flat background used to acquire the flat image(s) to be as close as possible to a true gray (for which all RGB components would have identical values)
`--dark-setup`, `--flat-setup`	optional configuration scripts to run before dark and flat acquisitions
`--dark-histogram`, `--flat-histogram`	optional; path to output file showing the histogram of dark and flat image pixel components; this gives a visual overview of the dark current variations as well as the variations in the flat image
`--no-interact`	normally the wizard waits for the user to setup the system before starting the dark and flat acquisitions; when this flag is used, the wizard does not wait for the user (nor does it open the created histogram html pages)

Example

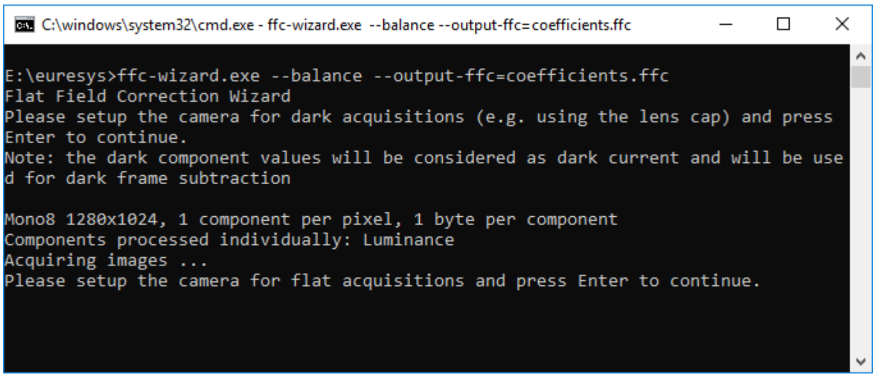
Here is an illustrated example that generates FFC coefficients (written to the file `coefficients.ffc`) using the white balance functionality. The command to run from a console window is the following:



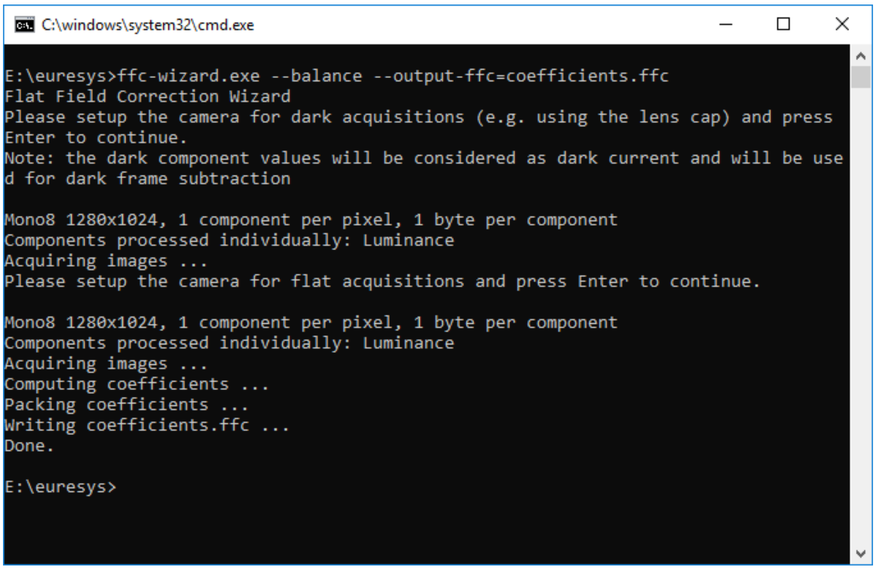
The program starts and tells you what to do:



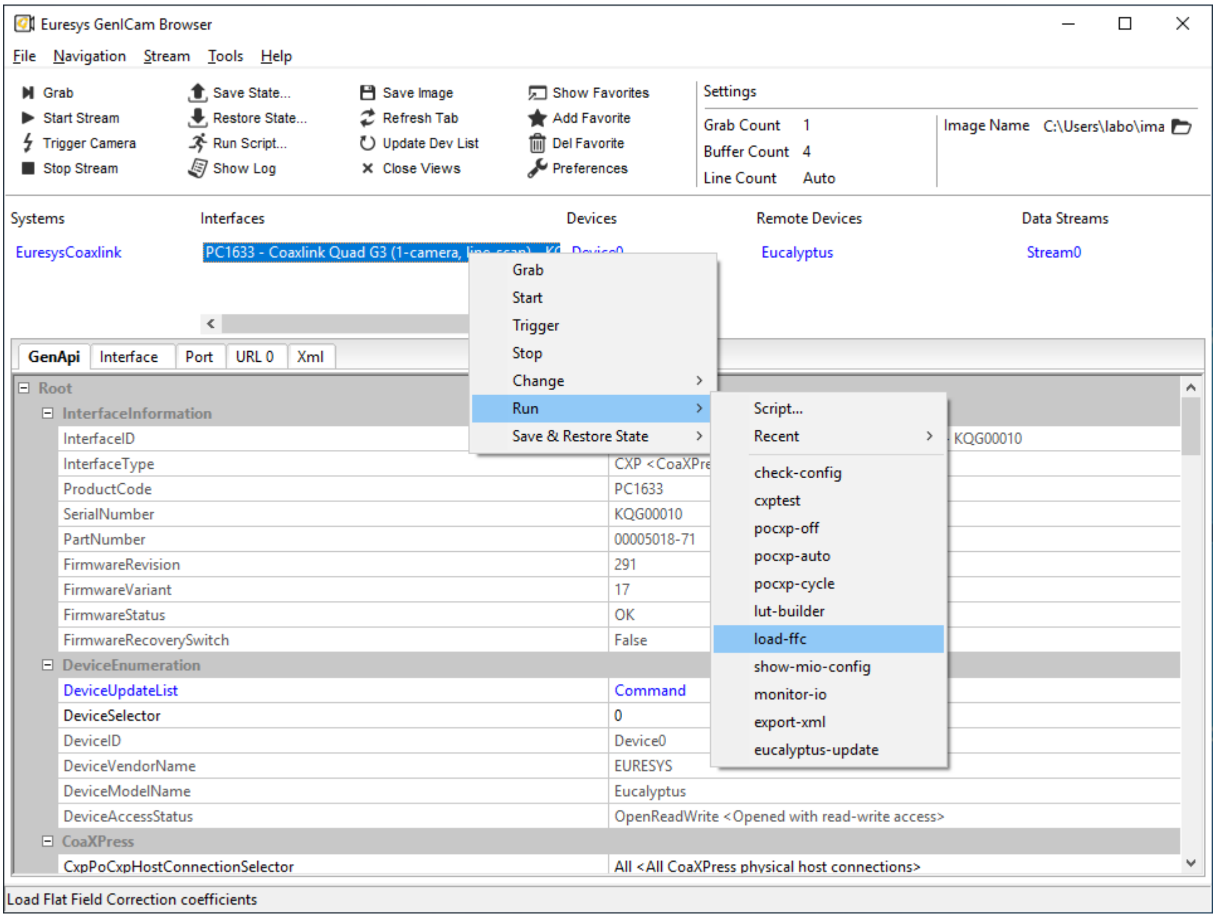
You can prepare your setup for the dark acquisitions and press Enter when you are ready. It will then acquire the series of dark images and display the instructions for the next step



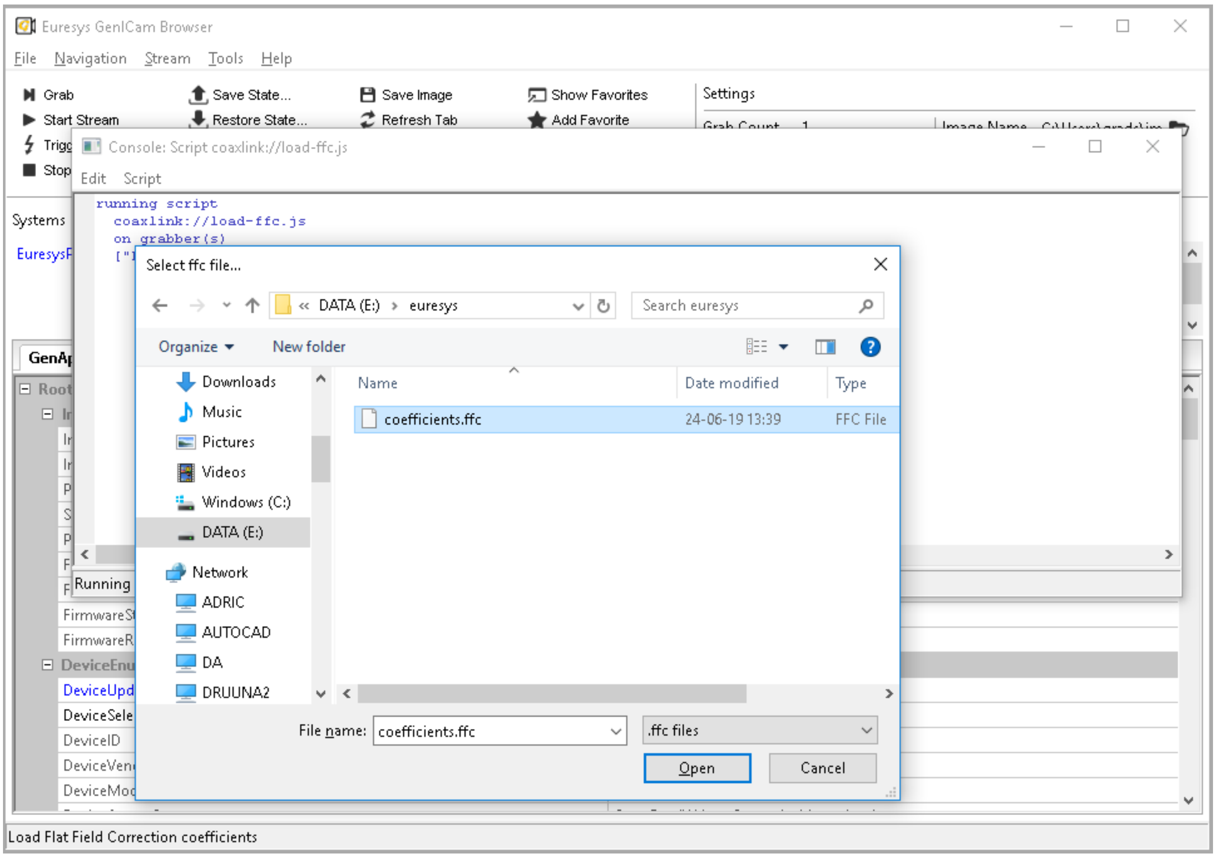
You can prepare your setup for the flat acquisitions and press Enter when you are ready. It will then acquire the series of flat images, compute the corresponding coefficients and write them to the file `coefficients.ffc`



Later on you can load the coefficients from the genicam-browser for example, by running the load-ffc script as follows:



Then you can select the previously created file coefficients.ffc



From that moment, the coefficients are loaded into the Coaxlink memory and the FFC processing is enabled.

Design

The calibration procedure as well as the packing of the coefficients is controlled by the main function `ffcWizard`.

`ffcWizard` tasks

Task	Done by function
acquiring a series of dark images to build an average dark image	`acquireImages`
acquiring a series of flat images to build an average flat image	`acquireImages`
computing the `gain` values for each pixel component	`computeGain`
using the dark pixel component values as `offset` values	`computeOffset`
packing `offset` and `gain` values into 16-bit little-endian values	`packCoefficients`
writing the packed coefficients into a binary file	`savePackedCoefficients`

Customization

The sample application already supports a few common pixel formats: `Mono`, `RGB`, `RGBa` and `Bayer`.

Limitation: to limit the complexity of the sample application, we consider (for pixel formats with several components per pixel) that all components have the same size. Supporting pixel formats with different component sizes is still possible by updating the functions `addImage` and `addComponents`.

To support a new pixel format (under the condition of the previous limitation), we need to modify two functions:

1. `Image::getComponentsPerPixel`, to return the number of components per pixel for the new format identified by its PFNC name
2. `Image::getComponentFilters`, to return a `std::vector` of `ComponentFilter` objects describing how the pixel components of the new format (identified by its PFNC name) are positioned in the image

The `ComponentFilter` objects are used to separate the processing of the different pixel components while evaluating the `Gain` and `Offset` values. For example, in `RGB` format, the FFC coefficients related to the `Red` components are computed using the `Red` components from the dark and flat images.

Please see the source code of `Image::getComponentFilters` for details about pixel component layout configuration.

11.5. Lookup Table Processing

WARNING
This function is only available for a restricted set of product and firmware variants combinations!

See also: "Firmware Variants per Product" on page 20 and search for *LUT* in the *Advanced Processing* column.

WARNING
This release of the Coaxlink driver provides lookup table processing for monochrome pixel formats exclusively.

Introduction to LUT Processing

This release of the Coaxlink driver provides lookup table processing for monochrome pixel formats exclusively.

See also: "Monochrome Lookup Table Processing" on the next page for a detailed description.

The Coaxlink driver provides four methods to define the content of lookup tables.

See also: "LUT Content Definition" on page 196

See also: "LUT Setup Procedure" on page 203 to setup lookup tables.

Monochrome Lookup Table Processing

Configurations

The following table lists all the available lookup table configurations for monochrome pixels:

Configuration	Input Pixel Format [PFNC]	Input bits	Output bits
M_8x8	Mono8, Raw	8	8
M_10x8	Mono10	10	8
M_10x10			10
M_10x16			16
M_12x8	Mono12	12	8
M_12x12			12
M_12x16			16

NOTE
Monochrome 8-bit pixels can be transformed into monochrome 8-bit pixels, monochrome 10-bit pixels can be transformed into monochrome 8-bit, 10-bit or 16-bit pixels and, monochrome 12-bit pixels can be transformed into monochrome 8-bit, 12-bit or 16-bit pixels.

Lookup Table Data Sets

A *lookup table data set* is defined as the set of data required to configure one lookup table for each component of a pixel. In the case of monochrome pixels, a lookup table data set includes only one single lookup table content.

The number of lookup table data sets that can be uploaded depends on the lookup table configuration:

Configuration	Data Sets
M_8x8	16
M_10x8, M_10x10, M_10x16	4
M_12x8, M_12x12, M_12x16	1

LUT Content Definition

The Coaxlink driver provides four methods to define the content of a lookup table.

Response Control Method

This method defines the transfer function of a lookup table by means of four parameters: Contrast, Brightness, Visibility and Negative.

The Contrast and Brightness parameters provide controls similar to the brightness and contrast controls of a television monitor.

The Visibility parameter provides control to smoothly reshape the transfer function to cover the full input range.

The Negative parameter allows transforming an image into its negative image.

Emphasis Method

This method defines the transfer function of a lookup table by means of two parameters: Emphasis and Negative.

It allows transforming an image using a power-law expression also known as γ – Gamma – function.

The Negative parameter allows transforming an image into its negative image.

Threshold Method

This method defines a double threshold transformation law by means of five parameters: SlicingLevel, SlicingBand, LightResponse, BandResponse, DarkResponse.

Table Method

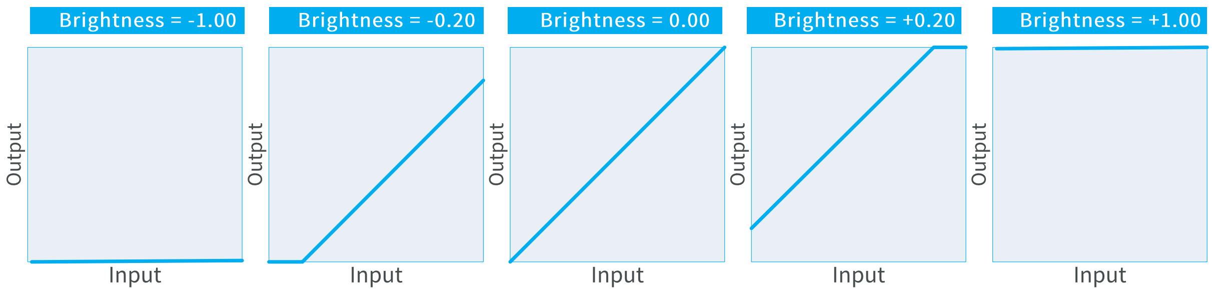
This method defines the transfer function of a lookup table in a tabular form.

Brightness Parameter

The Brightness parameter exclusively applies to the Response Control lookup table definition method.

It implements a control similar to the brightness control of a television monitor.

Brightness	Note
-1.00	Minimum value. Darkest output. The whole input range data gets transformed into the full black. This rule applies for any chosen Contrast value.
0.00	Default value. The mid-level input level of 0.5 is transformed as the same output level of 0.5. This is true for any value of the other parameters. Any increase in the brightness towards +1.00 results into a lighter output. Any decrease of the brightness towards -1.00 results into a darker output.
+1.00	Maximum value. Lightest output.The whole input range data gets transformed into the full white. This rule applies for any chosen Contrast value.



Effect of Brightness when all other controls are set to their default value:
Contrast = 1.00; Visibility = 0.00; Negative = FALSE.

Contrast Parameter

The Contrast parameter exclusively applies to the Response Control lookup table definition method.

It implements a control similar to the contrast control of a television monitor.

The slope of the transformation law is the gain, which is non-linearly controlled from the Contrast parameter.

Mathematically, the relationship is:

$$\text{Gain} = 10^{2 \times (\text{Contrast}-1)}$$

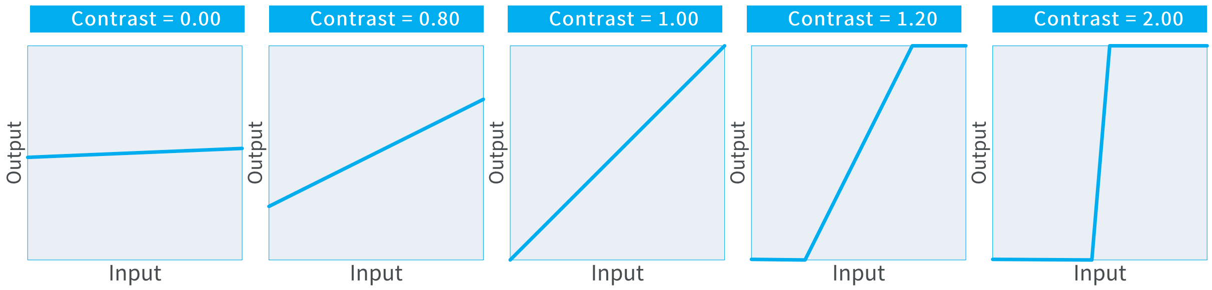
Contrast	Gain	Note
0.00	0.01	Min. `Contrast` value; smallest gain
1.00	1	Default `Contrast` value;unity gain
2.00	100	Max. `Contrast` value; largest gain

To achieve a required given gain, the contrast control should be set to:

$$\text{Contrast} = 1 + (\log_{10} \text{Gain})/2$$

If the required gain is expressed in decibels (dB):

$$\text{Contrast} = 1 + \text{Gain(dB)}/40$$



Effect of Contrast when all other controls are set to their default value:
Brightness = 0.00; Visibility = 0.00; Negative = FALSE

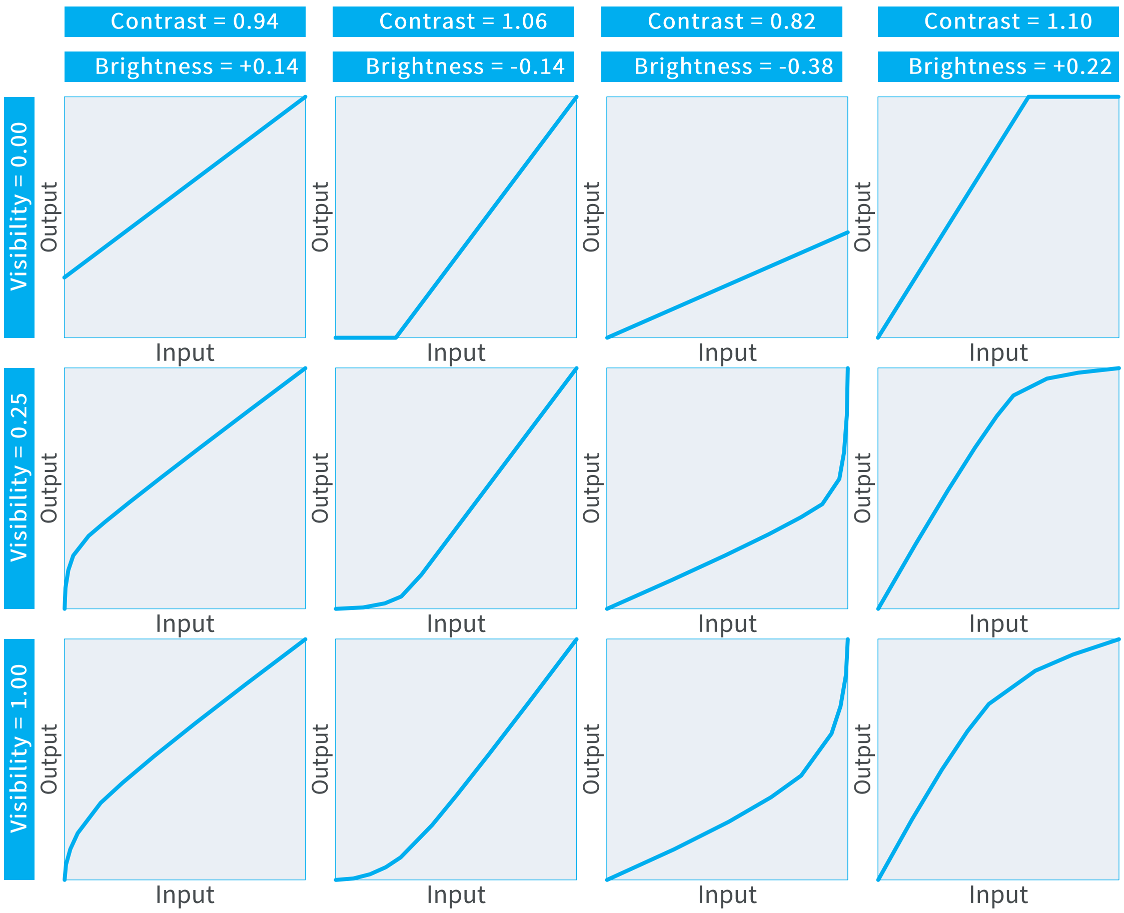
Visibility Parameter

The Visibility parameter exclusively applies to the Response Control lookup table definition method.

The operation of Contrast and Brightness parameters occasionally removes some part of the input dynamics. Very dark regions of the image can be transformed into full black, and become invisible. This holds true for very bright regions, clipping to full white.

The Visibility parameter has been created to smoothly reveal these hidden parts in the image.

Visibility	Gain	Note
0.00	0.01	Minimum and default value. This generates the piecewise linear transformation curves. Choosing values closer to +1 generates smoother curves.
1.00	100	Maximum value.



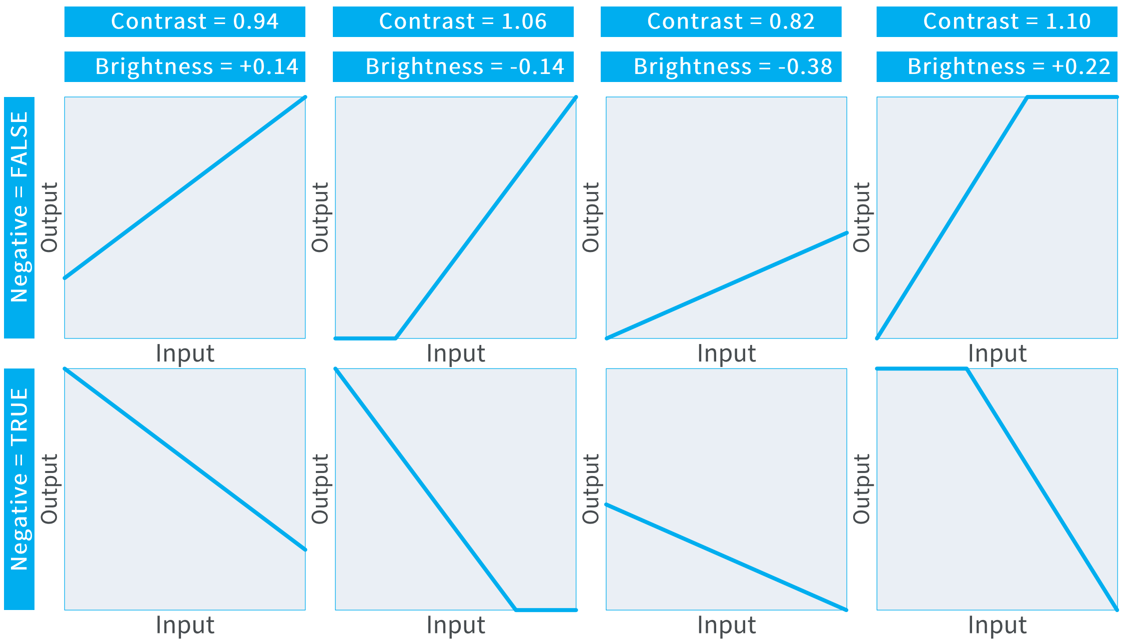
Effect of Visibility for typical values of Contrast and Brightness parameters assuming that Negative = FALSE

Negative Parameter

The Negative parameter applies to both the Response Control and the Emphasis lookup table definition methods.

This control allows transforming an image into its negative image, where the lightest areas of the image appear darkest and the darkest areas appear lightest.

Negative	Note
FALSE	Default value.
TRUE	The transformation table is mirrored around a vertical axis in the graphs. This swaps the black and white values, and gives rise to a photographic negative effect.



Effect of Negative for typical values of other controls

Emphasis Parameter

The Emphasis parameter exclusively applies to the Emphasis lookup table definition method.

It allows transforming an image using a power-law expression:

$$\text{Output} = \text{Input}^{\gamma}$$

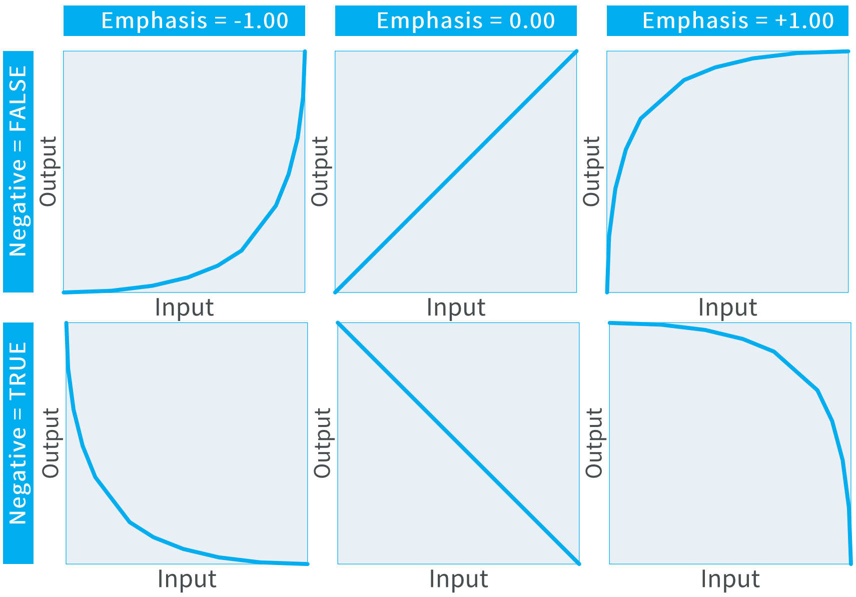
The γ – Gamma – exponent is mathematically linked to Emphasis by:

$$\gamma = 10^{-\text{Emphasis}}$$

Emphasis	Gamma	Note
1.00	0.1	Max. Emphasis value; smallest γ value
0.00	1	Default Emphasis value; linear law
-1.00	10	Min. Emphasis value; largest γ value

To achieve a required given γ, Emphasis should be set to:

$$\text{Emphasis} = -\log_{10}\gamma$$



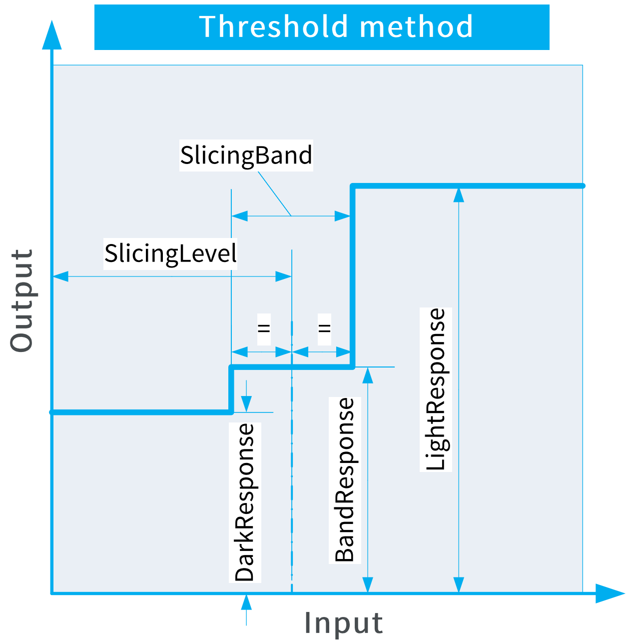
Emphasis effect for typical values of Emphasis and both values of Negative

Threshold Method Parameters

SlicingLevel, SlicingBand, LightResponse, BandResponse and DarkResponse parameters exclusively apply to the Threshold lookup table definition method.

As shown on the next figure, the parameters set defines a double threshold transformation law.

Parameter	Minimum Value	Default Value	Maximum Value
SlicingLevel	0.00	0.50	1.00
SlicingBand	0.00	0.50	1.00
LightResponse	0.00	0.75	1.00
BandResponse	0.00	0.50	1.00
DarkResponse	0.00	0.25	1.00



Double threshold transfer function

> **NOTE**
> SlicingLevel specifies the mean value of both thresholds in the input range.

LUT Setup Procedure

To setup the lookup table processing, proceed as follows:

1. Disable the lookup table
2. Define the lookup table configuration
3. Define the content of the lookup table
4. Upload the lookup table content into a specified lookup table data set
5. Enable the lookup table with a specified data set

Disabling the lookup table

To disable the lookup table:

- Set the LUTEnable feature to a Off.

Defining the lookup table configuration

To define the lookup table configuration, set the LUTConfiguration feature according to:

- The camera pixel type and bit depth
- The required output bit depth.

See also: "Monochrome Lookup Table Processing" on page 195 for configurations applicable to monochrome pixels.

NOTE
The lookup table configuration must be set prior to any other action.

Defining the lookup table content

See also: "LUT Content Definition" on page 196 for a description of the parametric and tabular methods used for defining a lookup table content.

NOTE
At least one lookup table set must defined.

Upload a lookup table content

To upload a lookup table content in one operation:

- Select a lookup table data set to access by assigning the appropriate value to the LUTSet feature. For instance Set1.
- Set the LUTIndex feature to 0.

- Write a string of LUTLength values to the LUTValue feature.

> **NOTE**
> The application may also selectively upload any individual lookup table entry or any block of consecutive lookup table entries.

Reading back a lookup table data set

To read back the lookup table data set in one operation:

- Select a lookup table data set to access by assigning the appropriate value to the `LUTSet` feature. For instance `Set1`.
- Set the LUTIndex feature to 0.
- Set the LUTReadBlockLength feature to the value returned by LUTLength.
- Get a string of LUTReadBlockLength values from the LUTValue feature.

> **NOTE**
> The application may also selectively read any lookup table entry individually or any block of consecutive entries.

Enabling the lookup table

To enable the lookup table:

- Set the LUTEnable feature to a value designating the lookup table data set to use.

Configuration Script Example

The following script is an example illustrating how to configure the lookup table for monochrome 8-bit to 8-bit operation and to define and upload 4 lookup table data sets using different lookup table definition methods.

```
function configure(g) {
    // Disable the lookup table
    g.StreamPort.set('LUTEnable', 'Off');
    // Configure the lookup table
    g.StreamPort.set('LUTConfiguration', 'M_8x8');

    // Build lookup table data set 1: response control
    g.StreamPort.set('LUTSet', 'Set1');
    require('coaxlink://lut/response-control')(g, { Contrast: 0.94
                                                , Brightness: 0.14
                                                , Visibility: 0.25
                                                , Negative: false });

    // Build lookup table data set 2: emphasis
    g.StreamPort.set('LUTSet', 'Set2');
    require('coaxlink://lut/emphasis')(g, { Emphasis: 0.5
                                        , Negative: true });

    // Build lookup table data set 3: threshold
    g.StreamPort.set('LUTSet', 'Set3');
    require('coaxlink://lut/threshold')(g, { SlicingLevel:  0.5
                                         , SlicingBand:   0.5
                                         , LightResponse: 0.75
                                         , BandResponse:  0.5
                                         , DarkResponse:  0.25 });

    // Build lookup table data set 4: table
    g.StreamPort.set('LUTSet', 'Set4');
    var i;
    for (i = 0; i < 256; ++i) {
        g.StreamPort.set('LUTIndex', i);
        g.StreamPort.set('LUTValue', String(255 - i));
    }
}
configure(grabbers[0]);
```

11.6. Bayer CFA Decoding

WARNING
This function is only available for a restricted set of product and firmware variants combinations!

See also: "Firmware Variants per Product" on page 20 and search for *CFA* in the *Advanced Processing* column.

Bayer CFA to RGB Conversion

The Bayer CFA decoder transforms the raw Bayer CFA data stream issued by the camera into an RGB color data stream.

The missing pixel components are reconstructed from the nearest components using one of the following interpolation methods:

- The *legacy interpolation method* computes the missing color components by applying exclusively the `Mean()` function.
- The *advanced interpolation method* computes the missing color components using the `Mean()` and the `Median()` functions.
  The advanced interpolation eliminates the aliasing effect on the highly contrasted sharp transitions in the image.

Functions Definitions

The `min()` function returns the lowest integer value from a set of 2 integer values.

The `max()` function returns the highest integer value from a set of 2 integer values.

The `mean()` function returns one integer value that represents the mean value of 2 integers. It is computed as follows:

Function	**`Mean(a,b)`**
Formula	(a + b + 1) >> 1

The `median()` function returns a set of two integer values that are the two median values of a set of four integers. It is computed as follows:

Function	**`Median(a,b,c,d)`**
Value 1 formula	`Min [ Max(a,b); Max(c,d) ]`
Value 2 formula	`Max [ Min(a,b); Min(c,d) ]`

Decoder operation

For each pixel of the source image, the CFA decoder computes two missing color components from surrounding pixels.

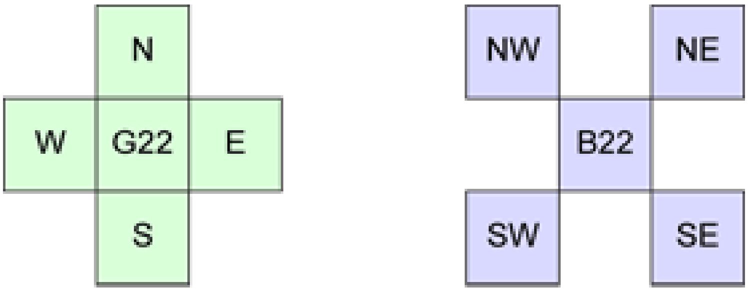
The text hereafter describes how the 4 central pixels located at positions 22, 32, 23 and 33 of a 4 x 4 Bayer CFA array are computed:

B11	G21	B31	G41
G12	R22	G32	R42
B13	G23	B33	G43
G14	R24	G34	R44

The relative positions of the surrounding pixels are identified by compass markings:

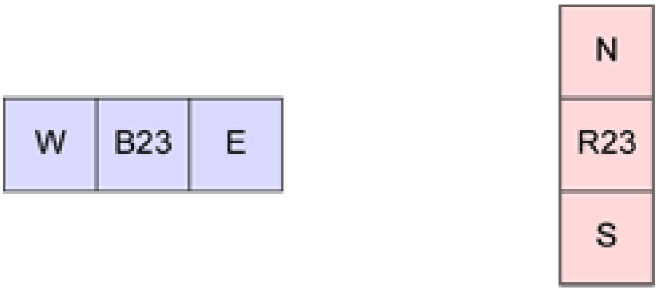
NW	N	NE
W		E
SW	S	SE

Formulas for position 22



Component	Method	Formula
R22	Legacy, Advanced	R22
G22	Legacy	Mean[Mean(N,S), Mean(W,E)]
	Advanced	Mean[Median(N, S, E, W)]
B22	Legacy	Mean[Mean(NW, SW), Mean(NE, SE)]
	Advanced	Mean[Median(NW, SW, NE, SE)]

Formulas for position 23

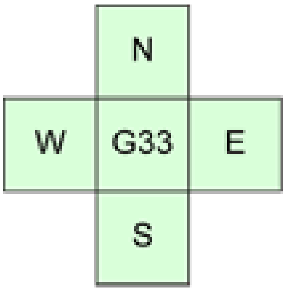


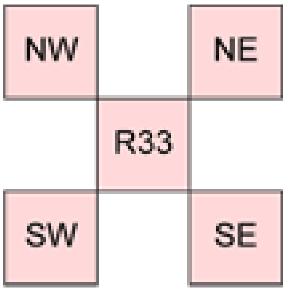
Component	Method	Formula
R23	Legacy, Advanced	Mean(N,S)
G23	Legacy, Advanced	G23
B23	Legacy, Advanced	Mean(W,E)

Formulas for position 32

Component	Method	Formula
R32	Legacy, Advanced	Mean(W,E)
G32	Legacy, Advanced	G32
B32	Legacy, Advanced	Mean(N,S)

Formulas for position 33





Component	Method	Formula
R33	Legacy	Mean[Mean(NW, SW), Mean(NE, SE)]
	Advanced	Mean[Median(NW, SW, NE, SE)]
G33	Legacy	Mean[Mean(N,S), Mean(W,E)]
	Advanced	Mean[Median(N, S, E, W)]
R33	Legacy, Advanced	B33

Using Bayer CFA Decoder

Prerequisites

1. *Frame grabber*
   Coaxlink card and firmware-variant with CFA decoder. Open "Firmware Variants per Product" on page 20 and search for *CFA* in the *Advanced Processing* column.
2. *Camera*
   a. Bayer CFA area-scan
   b. Up to 16384 pixels per line
   c. Having the color registration (GR, RG, GB, or BG) of the first two first transmitted pixels of the first transmitted line and the pixel bit depth (8-bit, 10-bit, 12-bit, 14-bit or 16-bit) correctly specified in the `PixelF` field of the CoaXPress image header.

> **WARNING**
> When the fields `Xoffs` and/or `Yoffs` are greater than 0, the camera must report an adapted `PixelF` value corresponding to the transmitted data!

Bayer to RGB Pixel Processing Configurations

When the Bayer CFA decoder is enabled:

- the "Pixel Component Unpacking" on page 69 control is inoperative, the Coaxlink card unpacks 10-bit, 12-bit or 14-bit pixels to lsb
- the "Pixel Component Re-Ordering" on page 71 feature allows to swap the Red and Blue components and deliver either BGR or RGB pixels

Input Pixel Format	FFC	RedBlueSwap	Output Pixel Format
Bayer**8	off | on	off	RGB8
		on	BGR8
Bayer**10pmsb	off | on	off	RGB10
		on	BGR10
Bayer**12pmsb	off | on	off	RGB12
		on	BGR12
Bayer**14pmsb	off | on	off	RGB14
		on	BGR14
Bayer16	off | on	off	RGB16
		on	BGR16

Enabling the Bayer CFA Decoder

In the Data Stream module, set the BayerMethod feature value to Legacy or Advanced according to the desired interpolation method.

Disabling the Bayer CFA Decoder

In the Data Stream module, set the BayerMethod feature value to Disable .

Bayer CFA Decoder Performance

Product	Firmware variant	Peak pixel processing rate
1633 Coaxlink Quad G3 **1633-LH Coaxlink Quad G3 LH**	1-camera	1,000 Mpixels/s
1635 Coaxlink Quad G3 DF	1-df-camera	1,000 Mpixels/s
3602 Coaxlink Octo	1-camera	2,000 Mpixels/s
	2-camera	1,000 Mpixels/s per stream
3603 Coaxlink Quad CXP-12	1-camera	2,000 Mpixels/s
3620 Coaxlink Quad CXP-12 JPEG	4-camera	500 Mpixels/s on stream0

NOTE
The peak pixel processing rate the Bayer CFA decoder approximately matches the performance of the DMA transfer when delivering RGB8 or BGR8 pixels. It is significantly lower than the maximum aggregated input data rate achievable by the board.

WARNING
The effective pixel processing rate may differ significantly from the peak pixel processing rate when the PCI Express available bandwidth is insufficient or when processing very short lines.

PCI Express Performance

PCI Express Interface	Sustainable PCIe data rate	RGB8	RGB10, RGB12, RGB14, RGB16
4-lane Rev 3.0 PCIe End-point	3,350 MB/s typical	~1,117 Mpixels/s	~558 MPixels/s
8-lane Rev 3.0 PCIe End-point	6,700 MB/s typical	~2,238 Mpixels/s	~1,117 Mpixels/s

NOTE
The PCI interface is not the bottleneck for 8-bit bit depths!

WARNING
For 10-, 12-, 14- and 16-bit bit depths, the sustainable data output rate is further limited by the performances of the PCI Express Interface on the Host PC.

Latency

The hardware CFA decoder performs on-the-fly conversion with a negligible latency when the data throughput is NOT limited by the available PCI Express bandwidth!

11.7. JPEG Encoding

Applies to:
QuadCXP12J

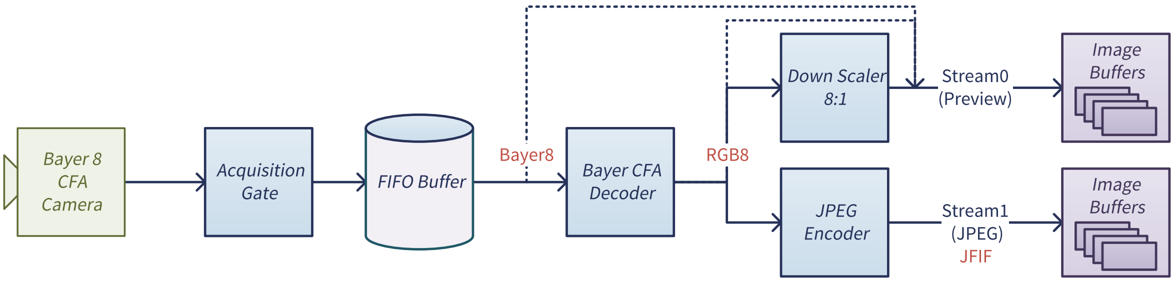
WARNING
This function is only available for a restricted set of product and firmware variants combinations!

See also: "Firmware Variants per Product" on page 20 and search for *JPEG* in the *Advanced Processing* column.

The **3620 Coaxlink Quad CXP-12 JPEG** is a four-connection CoaXPress CXP-12 frame grabber fitted with embedded CFA decoders and JPEG encoders.

The *4-camera* firmware variant implements 4 independent channels composed with:

- A 1-connection CoaXPress CXP-12 Host interface with PoCXP and Camera triggering capabilities,
- An acquisition gate that selects the frames to capture,
- A 512 MB FIFO buffer that provides a large elastic buffer,
- A Bayer CFA decoder that converts the Bayer8 camera data stream into an RGB8 stream using the advanced algorithm,
- A JPEG encoder including an RGB8 to YUV422 converter front-end and a baseline ISO/IEC 10918-1 compliant JPEG encoder,
- A down-scaler that reduces by a factor of 8 the RGB8 image resolution in both directions.



Camera requirements

- 8-bit Bayer CFA area-scan color camera,
- Image resolutions (H x V) : from 128 x 16 up to 5120 x 3840, with both width and height multiples of 8,
- CXP-1 up to CXP-12 CoaXPress interface.

Channel specifications

- Pixel processing rate: up to 250 Megapixels/second
- Output streams: 2 ('Preview' and 'JPEG')

- Encoder control: JPEG quality

Preview stream specifications

- Pixel formats: RGB8 or Bayer8
- Resolution: 'full' or '8:1' (for RGB8 only)

JPEG stream specifications

- JPEG control: Quality from 1 up to 100 (default: 75)
- Pixel format: CustomJFIF
- Resolution: 'full'
- Latency: 20 lines typical

11.8. Laser Line Extraction

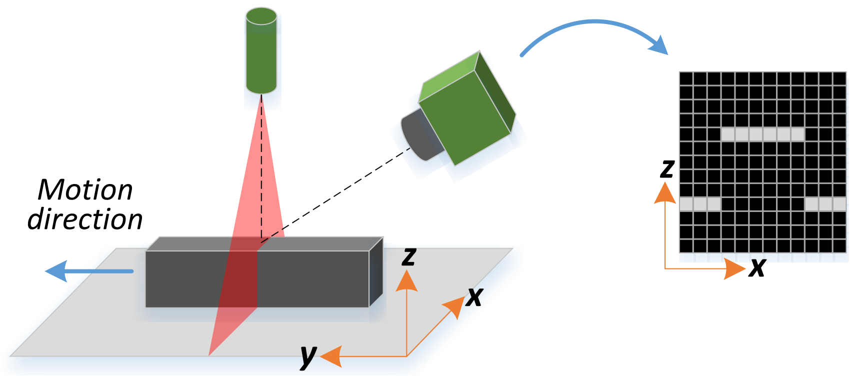
Applies to:
Quad3DLLE

WARNING
This function is only available for a restricted set of product and firmware variants combinations!

See also: "Firmware Variants per Product" on page 20 and search for LLE in the *Advanced Processing* column.

Introduction

In a laser-line triangulation system, a laser line (or any other “narrow light stripe” generation method) is projected on a 3D object. A camera, placed in another perspective than the laser, is then used to capture an image of that line, deformed by the shape of the object. The deformations of the line are a direct representation of the shape of the 3D object in the plane of the laser line. By scanning the object, that is making it move under the laser line and taking multiple images, you can reconstruct its 3D shape.



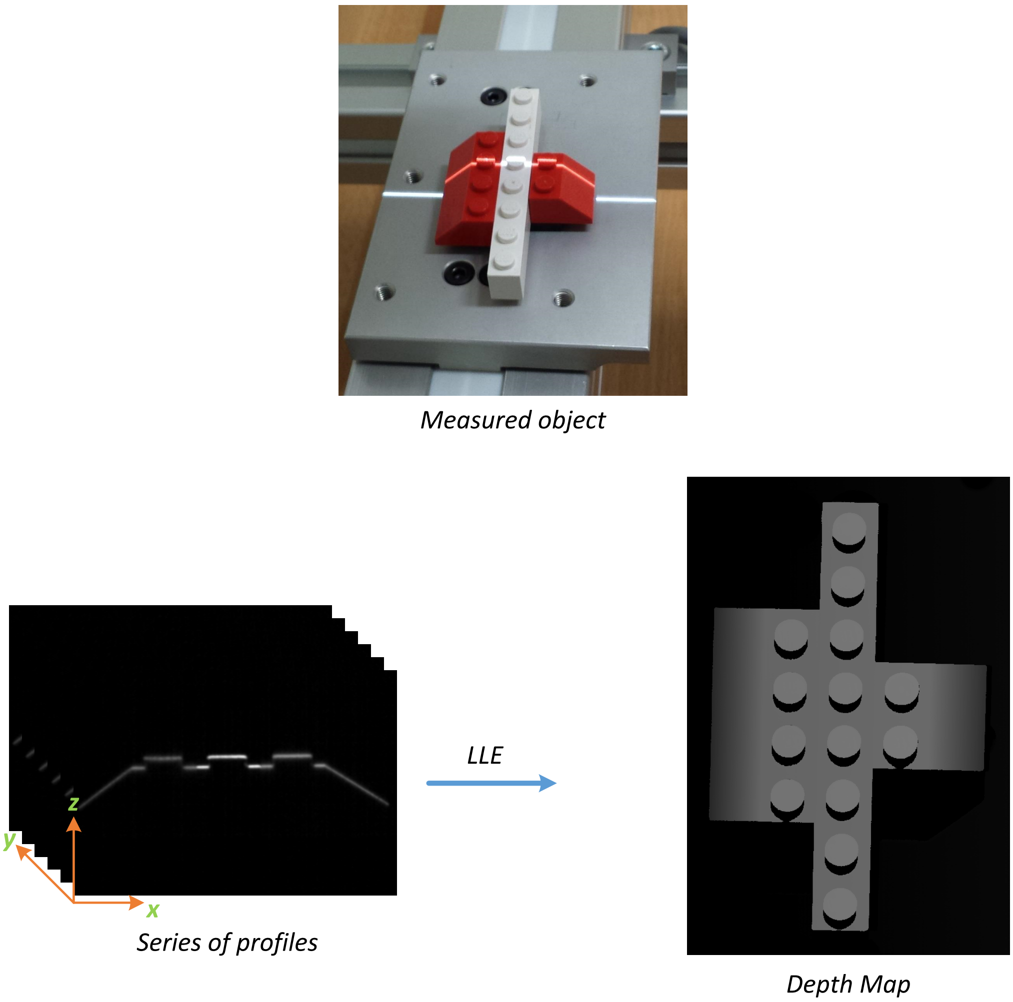
Simplified laser triangulation system

Laser Line Extraction Algorithms

A Laser Line Extraction (LLE) algorithm is required to create a *depth map* based on a sequence of profiles of the object captured by the camera sensor.

The objective of an LLE algorithm is to estimate the position where a laser line horizontally crosses a Region of Interest (ROI). The detection can be done by analyzing each column of a frame individually.

An LLE algorithm typically outputs a data array containing the vertical position of a detected laser line along of a ROI, i.e., each computed ROI produces a single data array.

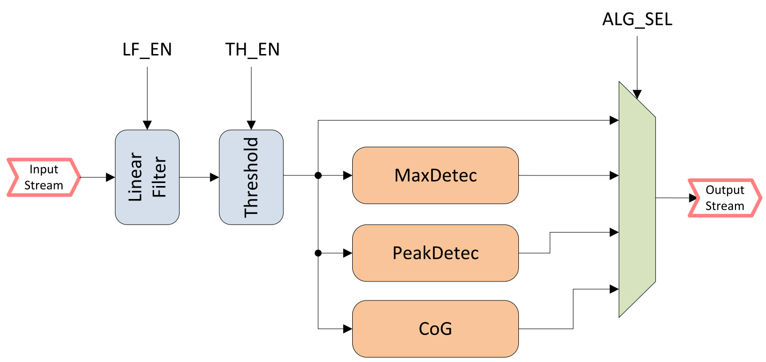


Depth map generation

LLE Processing Core Implementation

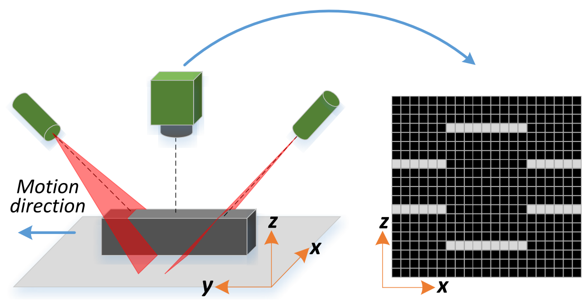
The LLE Processing Core is embedded in the **1637 Coaxlink Quad 3D-LLE** frame grabber. It can compute the *depth map* of a measured object with *zero host CPU usage* and *zero latency*.

It provides 3 algorithms for laser line extraction (*Maximum Detection*, *Peak Detection*, and *Center of Gravity*) as well as a pre-processing stage for filtering and thresholding.



Simplified block diagram of the LLE Processing Core

The LLE Processing Core is also capable to compute simultaneously up to 2 depth maps from a single input ROI. This feature is called *Dual Laser Line Extraction* (*Dual-LLE*). This feature is useful for applications using 2 laser lines projected on the same object.



Simplified dual laser triangulation system

When the Dual-LLE mode is activated, the LLE Processing Core input split the input ROI into two sub-ROIs, called LLE-ROI. Each LLE-ROI is independently processed by the LLE Processing Core resulting in a corresponding depth-map.

LLE Processing Core Characteristics

Absolute Maximum LLE-ROI XSize

- *All algorithms:* 8192 pixels

Max. Effective LLE-ROI YSize [pixels]

MaxDetec 8	MaxDetec 16	PeakDetec 11_5	PeakDetec 8_8	CoG 11_5	CoG 8_8
255	65535	2048	256	2048	256

Output Format [GenICam PFNC v2.1]

MaxDetec 8	MaxDetec 16	PeakDetec 11_5	PeakDetec 8_8	CoG 11_5	CoG 8_8
8-bit (integer)	16-bit (integer)	UQ11.5	UQ8.8	UQ11.5	UQ8.8

Output Pixel Format

MaxDetec 8	MaxDetec 16	PeakDetec 11_5	PeakDetec 8_8	CoG 11_5	CoG 8_8
Coord3D_C8	Coord3D_C16	Coord3D_C16	Coord3D_C16	Coord3D_C16	Coord3D_C16

Output Sub-pixel Resolution [pixel]

MaxDetec 8	MaxDetec 16	PeakDetec 11_5	PeakDetec 8_8	CoG 11_5	CoG 8_8
1	1	1/32	1/256	1/32	1/256

InvalidDataFlag Value

- *MaxDetec 8 algorithm:* 0x00
- *Other algorithms:* 0x0000

> **NOTE**
> This value identifies a non-valid result.

Valid Output Range

- *MaxDetec 8 algorithm:* 0x01 ~ 0xFF
- *Other algorithms:* 0x0001 ~ 0xFFFF

Supported Input Pixel Format

- *All algorithms:* Mono8

Number of Laser Lines

- *All algorithms:* 1 per LLE-ROI (up to 2 per input ROI)

Maximum Performance

- *All algorithms:* 2.500 megapixels/s
- *All algorithms:* 9,500 profiles/s from a 2048 x 256 or 4096 x 128 image
- *All algorithms:* 19,000 profiles/s from a 2048 x 128 image
- *All algorithms:* 38,000 profiles/s from a 1024 x 128 image
- *All algorithms:* 76,000 profiles/s from a 1024 x 64 image

NOTE
The above figures are based on the maximum bandwidth of a 4-connection CXP-6 CoaXPress link and considering that the dual LLE mode is activated.

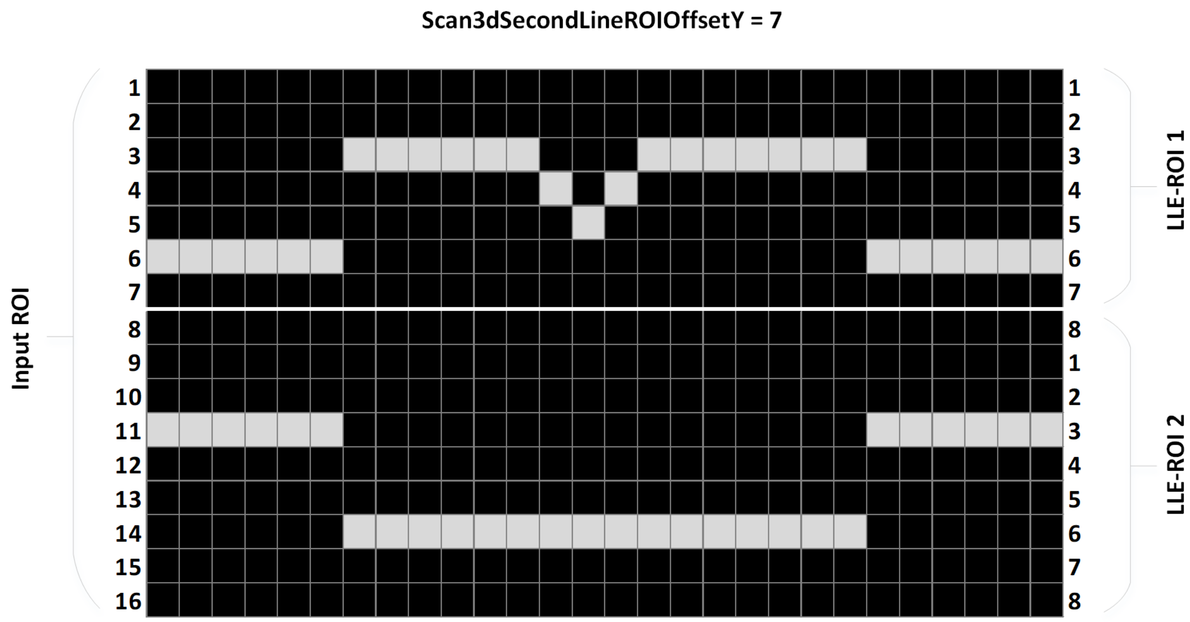
Available Output Types

- *All algorithms:*
  - Depth map
  - Raw camera image

WARNING
Not both outputs simultaneously!

Dual Laser-line Extraction

The Dual-LLE mode can be activated by setting the data stream feature Scan3dSecondLineROIOffsetY to a value different from 0. The feature Scan3dSecondLineROIOffsetY corresponds to an offset that defines where the input ROI will be split into two LLE-ROIs.

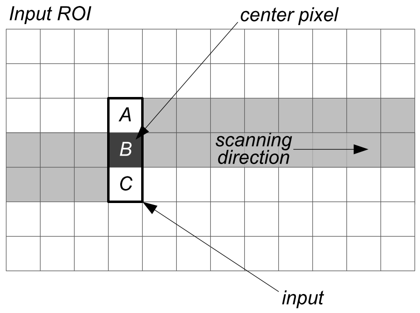


ROI Offset example

Linear Filter

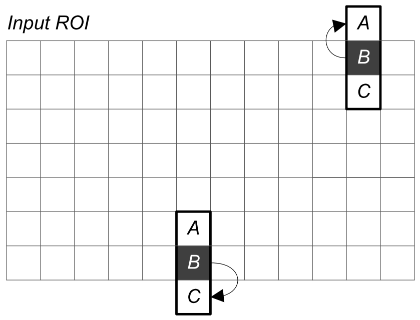
The *Linear Filter* module applies a convolution operator on a 1x3 sliding window directly on the data flow as it comes from the camera. The 3 elements of the convolution kernel (A, B, and C) are configurable accepting any positive integers where the sum of the 3 elements is a value between 1 and 512.

This figure illustrates the positioning of the convolution kernel elements within a given LLE-ROI:



Linear Filter kernel disposition

When an element of the kernel is located outside of the LLE-ROI boundaries (typical sliding window problem), its input pixel value is replaced by the central window pixel.



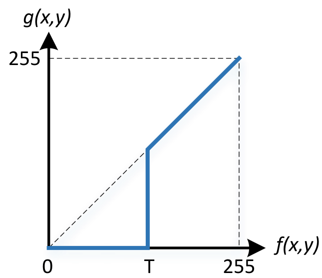
Linear Filter behavior at the LLE-ROI boundaries.

Coring Threshold

The coring is a non-linear operator that performs a simple segmentation technique, classifying pixels into two categories according to the following rule:

$$g\left(x,y\right)=\begin{cases} f(x,y), & f(x,y)>T \\ 0, & otherwise \end{cases}$$

T is the coring threshold.



Coring response

Maximum Detection

The Maximum Detection module can be configured to output a depth map in 8-bit or 16-bit data width. The difference between these two modes is the maximum effective LLE-ROI YSize supported by the Maximum Detection module. The 8-bit mode can represent heights of up to 255 Pixels and the 16-bit mode can represent heights of up to 65535 Pixels.

When a maximum intensity is detected in more than one pixel on a given LLE-ROI column, the Maximum Detection algorithm will indicate the one with the highest position.

InvalidDataFlag

An *InvalidDataFlag* is generated when:

- The detected line position is above the maximum LLE-ROI height.
- No MAX intensity is detected in an LLE-ROI column.

Peak Detection

The Peak Detection module produces a depth map represented by 16-bit fixed-point words. It can be configured with two precisions:

- *UQ11.5* in which 5-bit (LSB) represent the fraction part and 11-bit (MSB) the integer part. In this mode, the maximum effective LLE-ROI YSize supported by the Peak Detection module is almost 2048 Pixels.
- *UQ8.8* in which 8-bit (LSB) represent the fraction part and 8-bit (MSB) the integer part. In this mode, the maximum effective LLE-ROI YSize supported by the Peak Detection module is almost 256 Pixels.

When more than one peak is detected on a given LLE-ROI column, the Peak Detection module will indicate the position of the one where the corresponding f(x) pixel has the highest intensity. If more than one corresponding f(x) pixel have the same condition (highest intensity), then the one with highest position among them will be indicated.

InvalidDataFlag

An *InvalidDataFlag* is generated when:

- The detected line position is above the max LLE-ROI height supported.
- No line is detected in an LLE-ROI column.

Center of Gravity

The Center of Gravity module produces a depth map represented by 16-bit fixed-point words. It can be configured with two precisions:

- *UQ11.5* in which 5-bit (LSB) represent the fraction part and 11-bit (MSB) the integer part. In this mode, the maximum effective LLE-ROI YSize supported by the Center of Gravity module is almost 2048 Pixels.
- *UQ8.8* in which 8-bit (LSB) represent the fraction part and 8-bit (MSB) the integer part. In this mode, the maximum effective LLE-ROI YSize supported by the Center of Gravity module is almost 256 Pixels.

When more than one peak is detected on a given LLE-ROI column, the Center of Gravity module will indicate the position of the one where the corresponding f(x) pixel has the highest intensity. If more than one corresponding f(x) pixel have the same condition (highest intensity), then the one with highest position among them will be indicated.

InvalidDataFlag

An *InvalidDataFlag* is generated when:

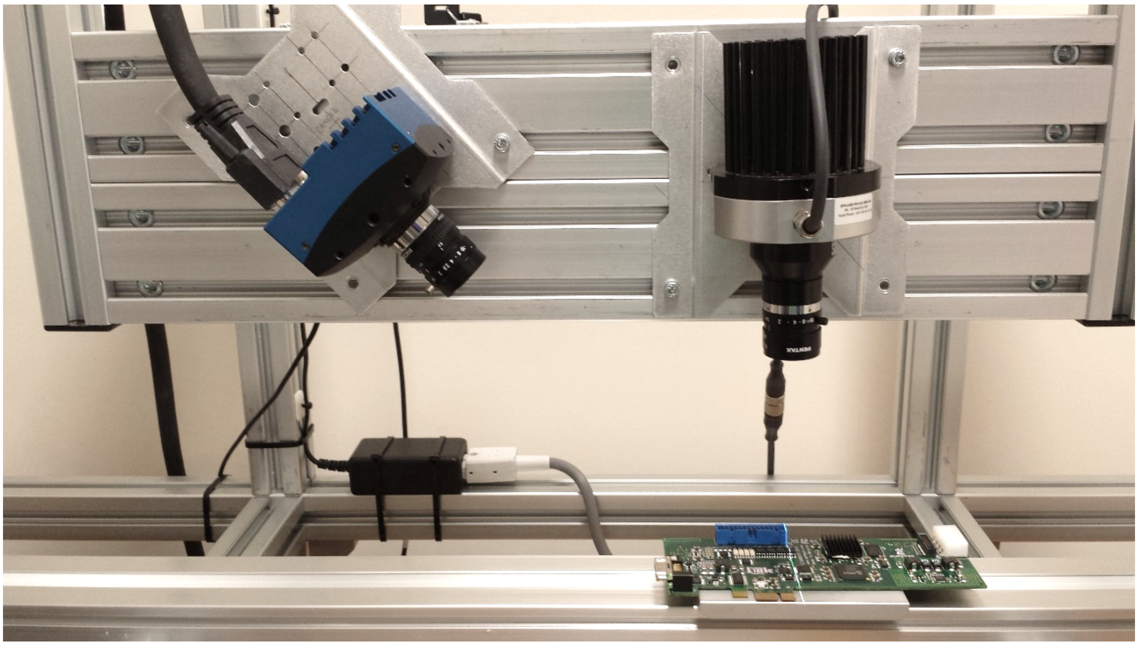
- The detected line position is above the max LLE-ROI height supported.
- An overflow occurs in an internal Sum. This condition can occurs when in a same LLE-ROI column a large number of *successive* pixels present intensities above the threshold level.
- No line is detected in an LLE-ROI column.

Use Case Example

Objective

In this example, the objective is to obtain a reliable measure of a PCB surface. For simplification purposes, the acquisition parameters are set using GenICam Browser and the camera is not controlled (free run). Other parameters like triangulation geometries, lenses, and laser color and power are not covered in this example.

This figure shows the laser line triangulation setup used in this example:

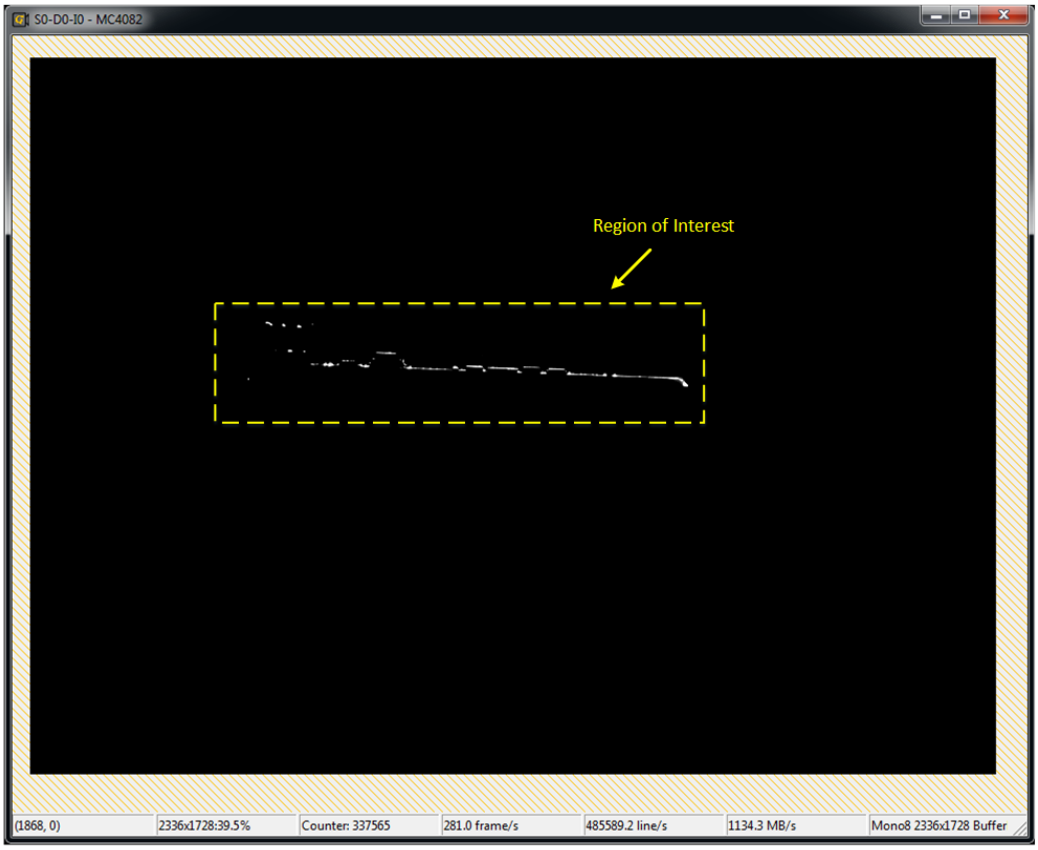


Laser line triangulation setup

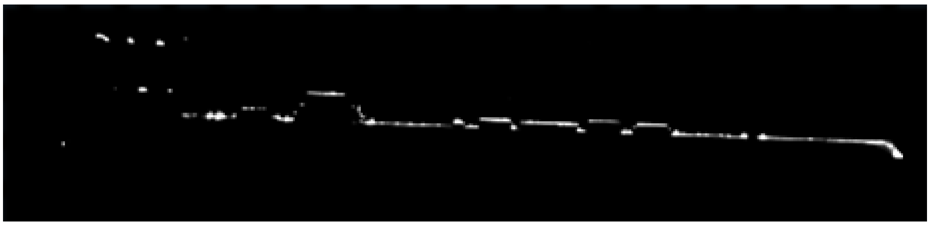
Defining the Input ROI

The defined Input ROI should cover the whole sensor region where the laser line variations occurs. The initial step is to set a full resolution ROI to verify the overall image.

The resulting frame can be seen in the following figure where it is possible to verify the effective region where the laser line variations occur.



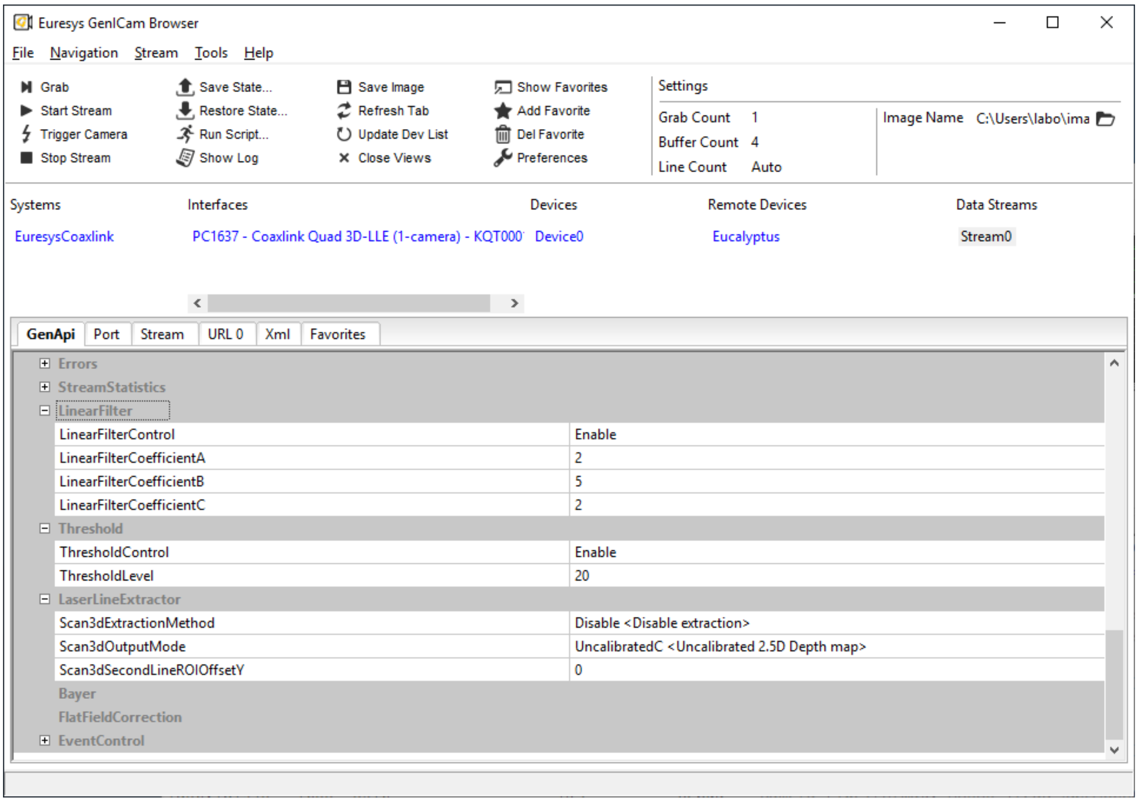
Resulting image (Full resolution ROI)



Input ROI Expanded View

Setting Filtering and Thresholding

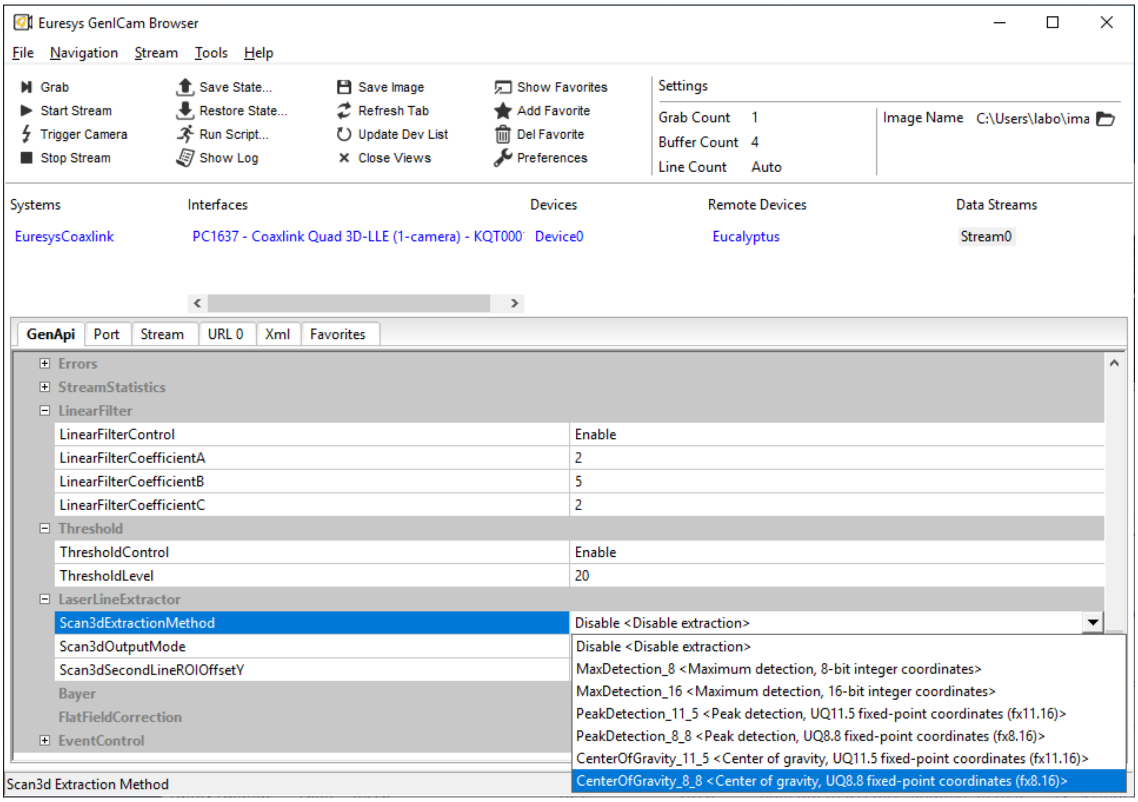
It is possible to activate a pre-processing stage in the LLE process. In the example shown below, the threshold level is set to 20 and the convolution filter kernel is set with A=2, B=5, and C=2. With these parameters, we will have a smoother image where any background noise below the threshold level is replaced by 0.



Filtering and Thresholding parameters

Defining the LLE Algorithm

The GenICam feature Scan3dExtractionMethod gives access to all available LLE Algorithms. The following figure shows how this feature is presented in GenICam Browser.



Laser Line Extraction options

The choice between the available algorithms depends on a number of factors related to the target application. We highlight here the main differences among the available algorithms.

LLE-ROI YSize

The number of effective vertical pixels in a LLE-ROI (YSize) is limited by the maximum integer value that can be represented by the output format of the chosen algorithm as described below:

- MaxDetection_8 (8-bit): 0xFF (255 pixels).
  - The value 0x00 is reserved to indicate an `InvalidDataFlag`.
- MaxDetection_16 (16-bit): 0xFFFF (65535 pixels).
  - The value 0x0000 is reserved to indicate an `InvalidDataFlag`.
- PeakDetection_11_5 (UQ11.5): 0xFFFF (almost 2048 pixels).
  - The value 0x0000 is reserved to indicate an `InvalidDataFlag`.
- PeakDetection_8_8 (UQ8.8): 0xFFFF (almost 256 pixels).
  - The value 0x0000 is reserved to indicate an `InvalidDataFlag`.
- CenterOfGravity_11_5 (UQ11.5): 0xFFFF (almost 2048 pixels).

- The value 0x0000 is reserved to indicate an `InvalidDataFlag`.

- CenterOfGravity_8_8 (UQ8.8): 0xFFFF (almost 256 pixels).
  - The value 0x0000 is reserved to indicate an `InvalidDataFlag`.

Algorithm Resolution

The position resolution of the available algorithms varies from 1 pixel up to 1/256 pixel as described below:

- MaxDetection_8 (8-bit): 1 pixel.
- MaxDetection_16 (16-bit): 1 pixel.
- PeakDetection_11_5 (UQ11.5): 1/32 pixel.
- PeakDetection_8_8 (UQ8.8): 1/256 pixel.
- CenterOfGravity_11_5 (UQ11.5): 1/32 pixel.
- CenterOfGravity_8_8 (UQ8.8): 1/256 pixel.

Algorithm Specificities

As mentioned previously, the choice of an LLE algorithm strongly depends on the target application. Following, a description of the main features of each LLE algorithm from the application point of view:

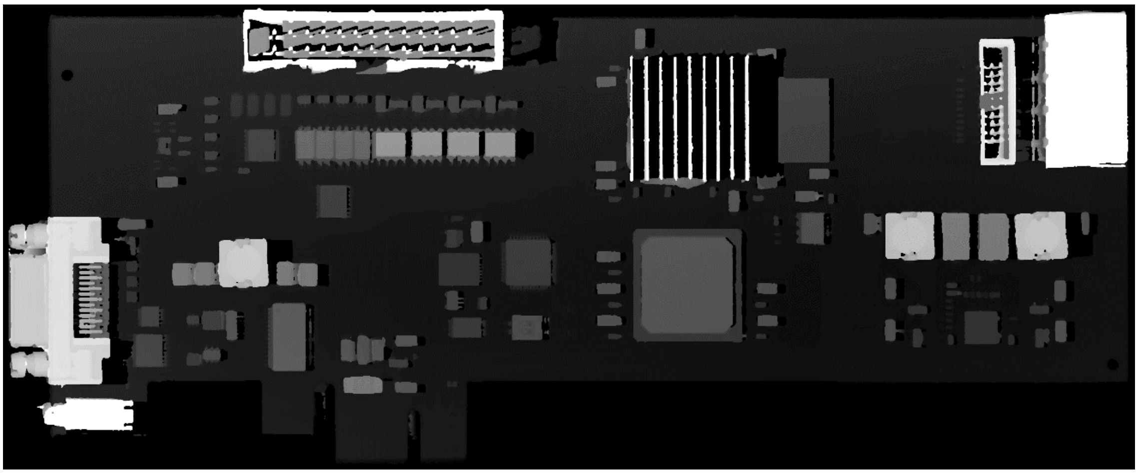
- MaxDetection_8 (8-bit): This version of the MaxDetection module with an 8-bit depth map output generates a very small amount of data which requires less computing performance from the 3D post-processing stage.
- MaxDetection_16 (16-bit): With its 16-bit integer depth map output, the version of the MaxDetection module can support very large LLE-ROI.
- PeakDetection: The two versions of the PeakDetection module offer a trade-off between sub-pixel resolution and maximum effective LLE-ROI size. The PeakDetection was designed to be insensitive to intensity-dependent biases.
- CenterOfGravity: The two versions of the CenterOfGravity module also offer a trade-off between sub-pixel resolution and maximum effective LLE-ROI size. The CenterOfGravity was designed to be robust to noisy inputs and wide lines.

Defining Scan Length and Buffer Size

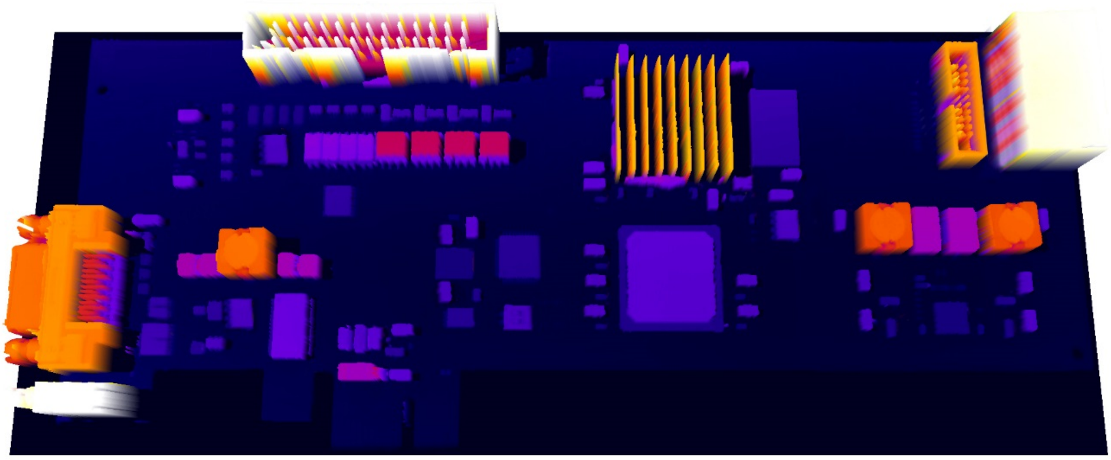
The ScanLength and BufferHeight will depend on the desired number of profiles to acquire from a given object. The size of the object being scanned, the camera frame rate, and the object movement speed are essential parameters to be defined along with the ScanLength.

Scanning Results

The resulting depth map of the measured PCB can be seen in the following figures.



Depth map example



Depth map example: 3D surface plot

Appendix

Fixed-point “Q-format” notation

The UQm.n system is a representation of fixed-point numbers in the Q-format where Q designates a number in the Q-forma notation, U preceding Q indicates an unsigned value, m indicates the number of bits of the integer portion of the number, and n designate the fraction portion the number. For instance, UQ11.5 is an unsigned fixed-point value represented by 16-bit words with 11 integer bits and 5 fractional bits. Other valid notation for UQ11.5 are fx11.16 and 0:11:5.

How to convert “UQ11.5” fixed-point values to floating-point

The UQ11.5 fixed-point values produced by the laser line extractor are 16-bit words representing values with 11 integer bits and 5 fractional bits.

To convert from this format to float, first convert the value to float, then divide by $2^5 = 32$.

```
float toFloat(unsigned short x) {
    float f = x;
    return f / 32;
}
```

How to convert “UQ8.8” fixed-point values to floating-point

The UQ8.8 fixed-point values produced by the laser line extractor are 16-bit words representing values with 8 integer bits and 8 fractional bits.

To convert from this format to float, first convert the value to float, then divide by $2^8 = 256$.

```
float toFloat(unsigned short x) {
    float f = x;
    return f / 256;
}
```

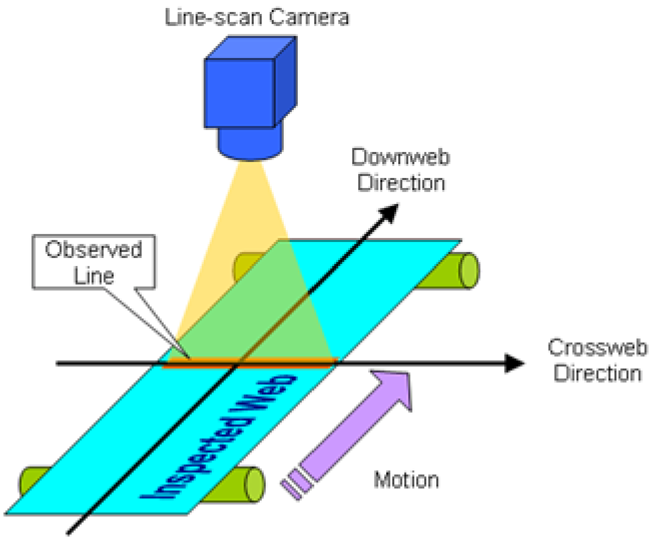
11.9. Line Scan Acquisition

WARNING
This function is only available for a restricted set of product and firmware variants combinations!

See also: "Firmware Variants per Product" on page 20 and search for *", line-scan"* suffix in the *Firmware Variant* column.

Line Scan Acquisition Principles

Line Scan Imaging



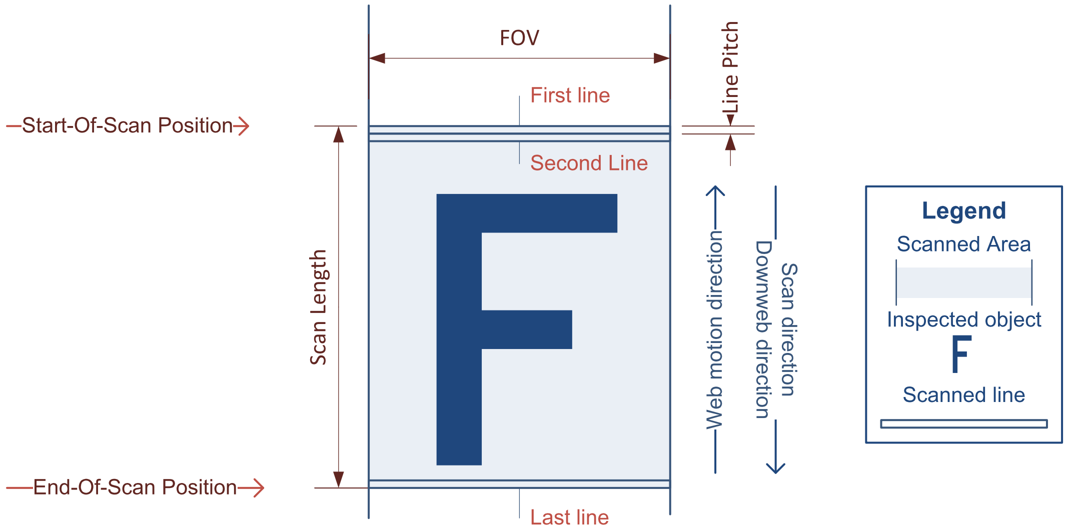
Typical line-scan imaging system

The expression “Line-scan imaging” designates machine vision applications where 2-D images are obtained by the combination of successive image lines captured from a 1-D imaging device that moves relatively to the object.

In line-scan imaging:

- The imaging device is often, but not necessarily, a line-scan camera.
- The inspected object is often a continuous web, it can also be discrete objects having fixed or variable size.
- The inspected web moves relatively to the camera. The motion speed during the acquisition can be fixed or variable.
  - The *cross-web direction* or *transverse direction* is the axis on the web plane that is observed by the camera.
  - The *down-web direction* or *axial direction* is the motion direction of the inspected web relatively to the camera.

Scanning Area



Scanning area definitions

The scanning area is a 2-D area on the web having a width equal to FOV and a length equal to Scan Length.

In the cross-web direction (horizontal direction in the above drawing), the scanning area is delimited by the field of view – FOV – of the camera.

In the down-web direction (vertical direction in the above drawing), the scanning area is delimited by the start-of-scan and the end-of-scan positions. The line pitch is determined by the ratio between the web speed and the camera line rate.

The *field of view – FOV* – of a line-scan camera is determined only by the optical setup and the sensor geometrical properties.

The *start-of-scan position* is a position on the web corresponding to the scan-line boundary preceding the first acquired line.

The *end-of-scan position* is a position on the web corresponding to the scan-line boundary following the last acquired line.

Most of the line-scan cameras are delivering a single row of pixels every camera cycle. Consequently, *multiple camera cycles* are necessary to build-up the object image.

Pixel Aspect Ratio Control

Unlike area-scan imaging, line-scan imaging allows the application to control the image pixel aspect ratio.

In the large majority of cases, the imaging application requires a constant, and preferably a 1:1 image pixel aspect ratio.

The cross-web pitch being locked by the sensor pitch and the optical magnification factor, the image pixel aspect ratio is controllable only through the line pitch control.

The following table summarizes the methods providing a constant line pitch that are applicable with Coaxlink:

Method Name	Description
VCR	***Variable Camera cycle.*** The *web speed is variable* and the *camera cycle rate is kept proportional to the web speed*. The frame grabber builds the object image by capturing all the successive lines delivered by the camera.
CCC	***Constant Camera Cycle.*** The camera operates at a constant cycle rate and the frame grabber captures all the successive lines delivered by the camera.

The VCR method requires:

- A *motion encoder* for measuring the web speed.
- A *real-time processing* of the motion encoder events to build a camera trigger at a rate that is *proportional* to the motion encoder events rate.

Having a proportional rate can be achieved by a *divider tool* or a *multiplier/divider tool*:

- The divider tool decimates the input rate by an integer value, it delivers 1 out of N incoming events.
- The multiplier/divider tool enables fine control of the image pixel aspect ratio by allowing any rate conversion ratio value – RCR – in the range 0.001 to 1000 with an accuracy better than 0.1% of the RCR value.

Image Acquisition with Line Scan Imaging Devices

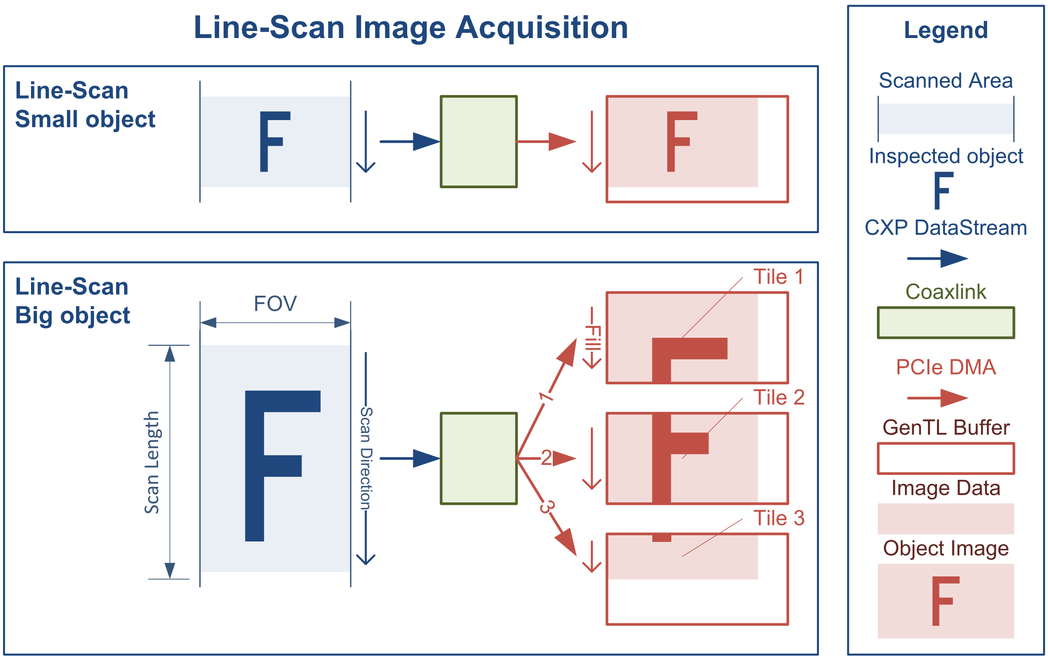


Image capture of small and large objects

For the transmission on the CoaXPress link, (most of) the line-scan cameras use *one CoaXPress Image Data Stream*.

Regarding the delivery methods of the image data, two cases are to be considered:

- For small objects, the object image data are delivered into a *single GenTL buffer*.
- For big objects, the object image data are delivered into *multiple GenTL buffers*.

In both cases, the image data are delivered through a single PCIe DMA channel and the transmission latency through Coaxlink is low: “one image line”.

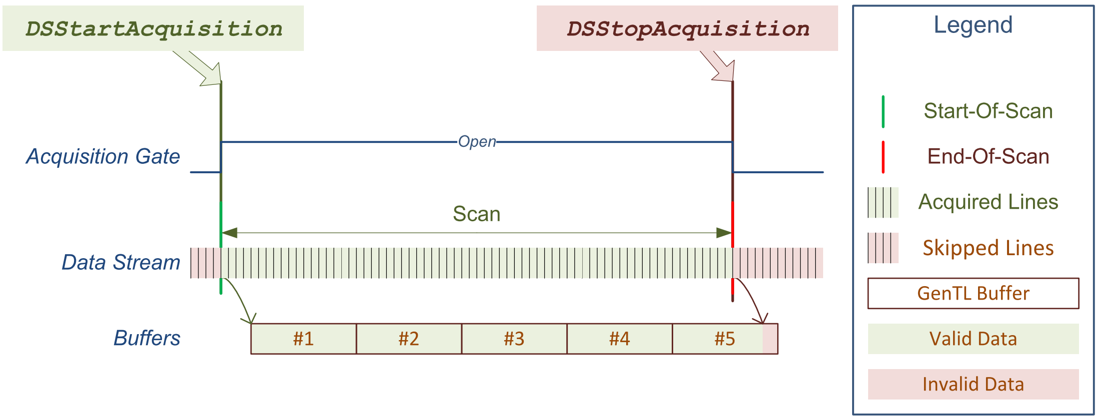
GenTL Buffer Filling Rules – Line-scan cameras

In line-scan imaging, GenTL buffers are filled according to the following rules:

- The first acquired line data of a scan is, by default, stored at the beginning of a new buffer. When vertical image flipping is enabled by setting StripeArrangement to Geometry_1X_1YE, the first acquired line data of a scan is stored at the location of the last full line of a new buffer.
- A buffer contains an integer number of image lines data.
- When the remaining space of a buffer is not sufficient to store an image line data, the acquisition continues into a new buffer and the filled buffer is made available to the application for processing.
- When the last line data of a scan is acquired, the last buffer, possibly partially filled, is made available to the application for processing.

Line Scan Acquisition Use cases

Scanning of continuous objects



Scanning of continuous objects

This case applies to the image scanning of **continuous objects**.

The Coaxlink acquisition controller is configured as follows:

- StartOfScanTriggerSource = Immediate.
- EndOfScanTriggerSource = StopScan.

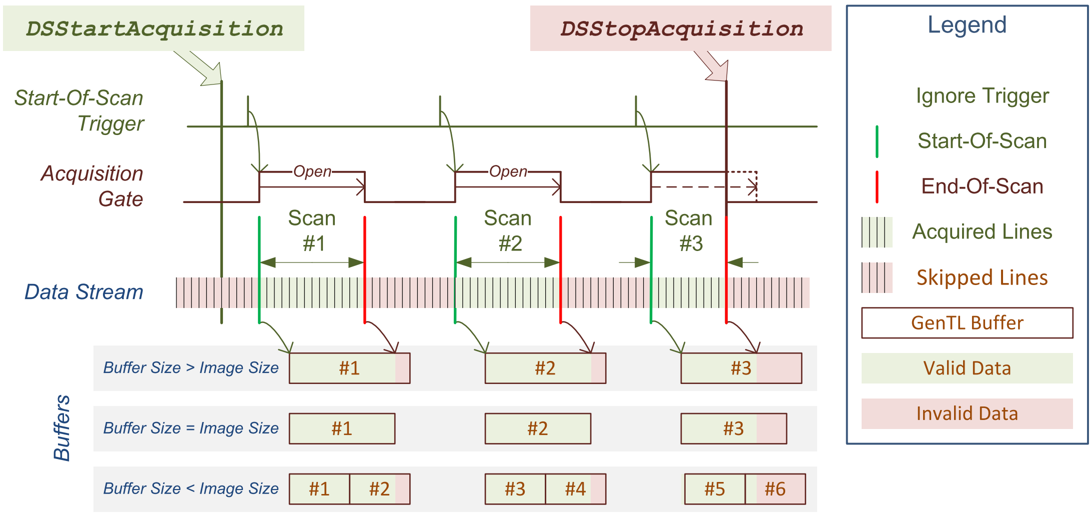
When the `DSStartAcquisition` function is called, the scanning starts at the next line boundary.

The acquisition gate closes when the application calls the `DSStopAcquisition` function.

Depending on the allocated buffer size and the scanning duration, the object image fits in a single buffer or requires multiple buffers.

Each buffer is delivered to the application as soon as it is filled. The last buffer, likely partially filled, is delivered as soon as the last image data are written.

Fixed-length scanning of discrete objects



Scanning of discrete objects with a common scan length

The Coaxlink acquisition controller is configured as follows:

- StartOfScanTriggerSource = StartScan or any applicable I/O Toolbox event output, for instance: LIN1.
- EndOfScanTriggerSource = ScanLength.
- ScanLength is any positive number representing the number of lines required to capture the object image entirely.

When the `DSStartAcquisition` function is called, the start-of-scan trigger of the acquisition controller is armed.

Then, the acquisition controller waits for the first occurrence of a valid start-of-scan trigger event.

A valid start-of-scan event can be generated:

- By the application using a StartScan command.
- By the selected hardware event source, if specified by StartOfScanTriggerSource.

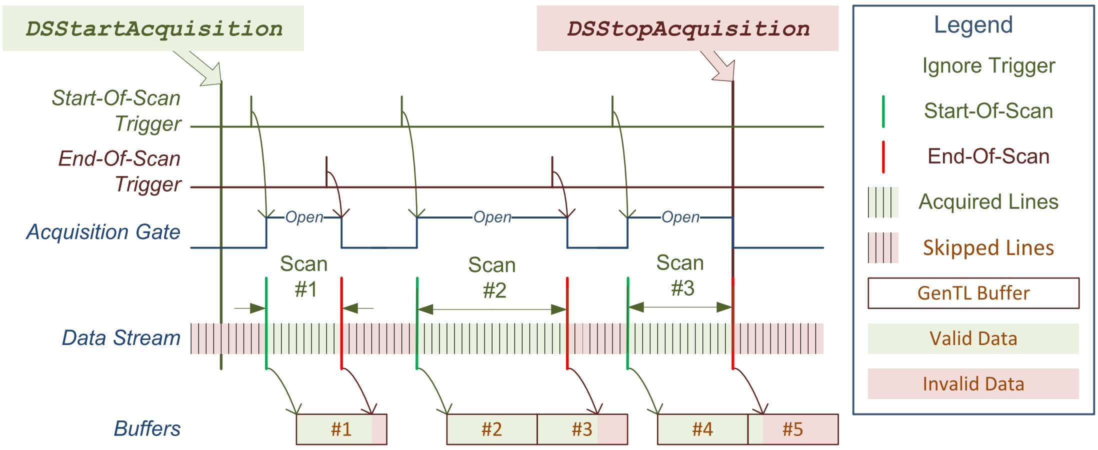
The acquisition controller ignores any Start-of-Scan trigger event while a scanning is in progress.

The acquisition gate opens at the first line boundary following a start-of-scan event.

The acquisition gate closes automatically after the specified number of lines have been acquired or anticipatively when the application calls the `DSStopAcquisition` function.

Depending on the allocated buffer size, the object image fits in a single buffer or requires multiple buffers. Each buffer is delivered to the application as soon as it is filled. At the end-of-scan, partially filled buffers are immediately delivered. The following image acquisition always begins with a new buffer.

Variable-length scanning of discrete objects



Scanning of discrete objects requiring a variable scan length

The Coaxlink acquisition controller is configured as follows:

- StartOfScanTriggerSource = StartScan or any applicable I/O Toolbox event output, for instance: LIN1.
- EndOfScanTriggerSource = StopScan or any applicable I/O Toolbox event output, for instance: LIN2.

When the `DSStartAcquisition` function is called, the start-of-scan trigger of the acquisition controller is armed.

Then, the acquisition controller waits for the first occurrence of a valid start-of-scan trigger event.

A valid start-of-scan event can be generated:

- By the application using a StartScan command.
- By the selected hardware event source, if specified by StartOfScanTriggerSource.

The acquisition controller ignores any Start-of-Scan trigger event while a scanning is in progress.

The acquisition gate opens at the first line boundary following a start-of-scan event.

The acquisition gate closes at the first line boundary following a valid end-of-scan event or immediately when the application calls the `DSStopAcquisition` function.

A valid end-of-scan event can be generated:

- By the application using a StopScan command.
- By the selected hardware event source, if specified by EndOfScanTriggerSource.

The acquisition controller ignores any End-of-Scan trigger event when no scanning is in progress.

Depending on the allocated buffer size, the object image fits in a single buffer or requires multiple buffers. Each buffer is delivered to the application as soon as it is filled. At the end-of-scan, partially filled buffers are immediately delivered. The next image acquisition always begins with a new buffer.

11.10. C2C-Link

C2C-Link Interconnections

The C2C-Link is a hardware communication medium allowing a single *C2C-Link master device* to reliably share trigger events with multiple *C2C-Link slave devices*.

In area-scan applications, the C2C-Link Interconnect:

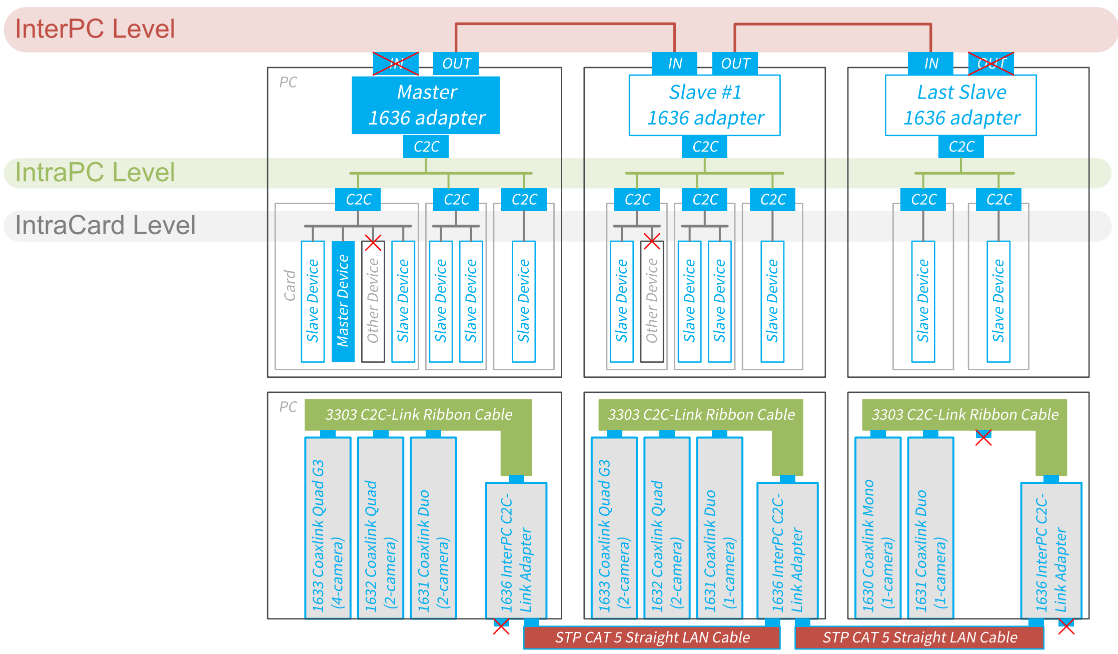
- allows to share up to 2 triggers: a CIC Cycle Trigger (mandatory) and oneI/O Toolbox event (optional).
- allows any C2C-Link device to indicate to the C2C-Link master device that it is not able to accept a trigger.

In line-scan applications, the C2C-Link Interconnect allows to share up to 3 triggers: a CIC Cycle Trigger (mandatory) and two I/O Toolbox events (optional).

The C2C-Link is scalable: it may interconnect devices belonging to the same Coaxlink card, or to different cards in the same PC or to different cards in different PCs.

A C2C-Link interconnection may combine up to three interconnection levels:

- The *IntraCard Level* interconnects 2 or more C2C-Link devices belonging to the same card using FPGA internal resources.
- The *IntraPC Level* interconnects C2C-Link devices across two or more cards of the same PC. It requires one accessory cable such as the **3303 C2C-Link Ribbon Cable** or a custom-made C2C-Link cable for each PC.
- The *InterPC Level* interconnects C2C-Link devices across two or more PCs. It requires one **1636 InterPC C2C-Link Adapter** for each PC and one RJ 45 CAT 5 STP straight LAN cable for each adapter but the last one.



C2C-Link configuration example using InterPC, IntraPC and IntraCard levels

C2C-Link Electrical Specification

Definitions

Trigger delay

Propagation delay of the trigger signal from the master device to a slave device. This delay is composed of the propagation delay inside electronic devices (FPGA, adapter...) and interconnection cables.

For cables, the delay is proportional to the cable length, typically: 5 ns/m.

Trigger delay skew

Dispersion of the trigger delay values across all the devices belonging to a C2C-Link.

Trigger delay jitter

Variation of the trigger delay depending on external factors such as temperature, signal noise...

Trigger rate

Rate of occurrence of repetitive trigger events. The reciprocal value (1/Trigger rate) is the minimum time interval required between consecutive triggers.

IntraCard C2C-Link Interconnection Level

Parameter	Min.	Typ.	Max.	Units
Trigger delay jitter			5	ns
Trigger rate	0		2.5	MHz

IntraPC C2C-Link Interconnection Level

Parameter	Min.	Typ.	Max.	Units
Cable connectors count (including 1636 adapter if any)	2		4	-
Cable length	–		0.6	m
Trigger delay jitter	5		10	ns
Trigger rate	0		2.5	MHz

InterPC C2C-Link Interconnection Level

The following specification targets applications where the *highest trigger rate* is required:

Parameter	Min.	Typ.	Max.	Units
1636 adapters count	2		10	-
Adapter-to-adapter LAN cable length	0.3		100	m
Cumulated adapter-to-adapter LAN cable length	0.3		100	m
Trigger delay jitter	10	20	40	ns
Trigger rate	0		2.5	MHz

The following specification targets applications where the *longest reachable distance* is required:

Parameter	Min.	Typ.	Max.	Units
1636 adapters count	2		32	-
Adapter-to-adapter cable length	0.3		1200	m
Cumulated adapter-to-adapter cable length	0.3		1200	m
Trigger delay jitter			1	µs
Trigger rate	0		200	kHz

NOTE
The maximum trigger rate specification can be extrapolated for intermediate distances between 100 m and 1200 m assuming that the *length x frequency* product is constant: in this case 250 [m. MHz].

Trigger Propagation Delays

The propagation delay of the Trigger signals from the master device to a slave device can be roughly estimated by adding the typical delays encountered in each segment of the signal path.

Typical Delay values per C2C-Link segment

Delay Element	Min.	Typ.	Max.	Units
(1) IntraPC interconnection (Whole IntraPC C2C-Link interconnect including FPGA I/O and IntraPC cable)	0	5	10	ns
(2) **1636 InterPC C2C-Link Adapter** – IntraPC-to-InterPC delay		15		ns
(3) **1636 InterPC C2C-Link Adapter** – InterPC-to-IntraPC delay		20		ns
(4) **1636 InterPC C2C-Link Adapter** – InterPC-to-InterPC delay		0		ns
(5) InterPC LAN cable		5		ns/m

Example 1 – IntraPC Configuration

For an IntraPC only configuration there is only one delay element to consider: (1)

Parameter	Min.	Typ.	Max.	Units
(1) IntraPC interconnection	0	5	10	ns
Total Trigger Delay	0	5	10	ns

Example 2 – 3-adapter InterPC Configuration; 20 m+20 m LAN cable

This configuration is composed of 3 Intra-PC segments. For devices belonging to the same IntraPC segment as the Master device, there is only one element to consider.

Parameter	Min.	Typ.	Max.	Units
(1) IntraPC interconnection	0	5	10	ns
Total Trigger Delay for devices of the Master InterPC segment	0	5	10	ns

For devices belonging to the same IntraPC segment as the Slave1 adapter, there are 5 delay elements to consider:

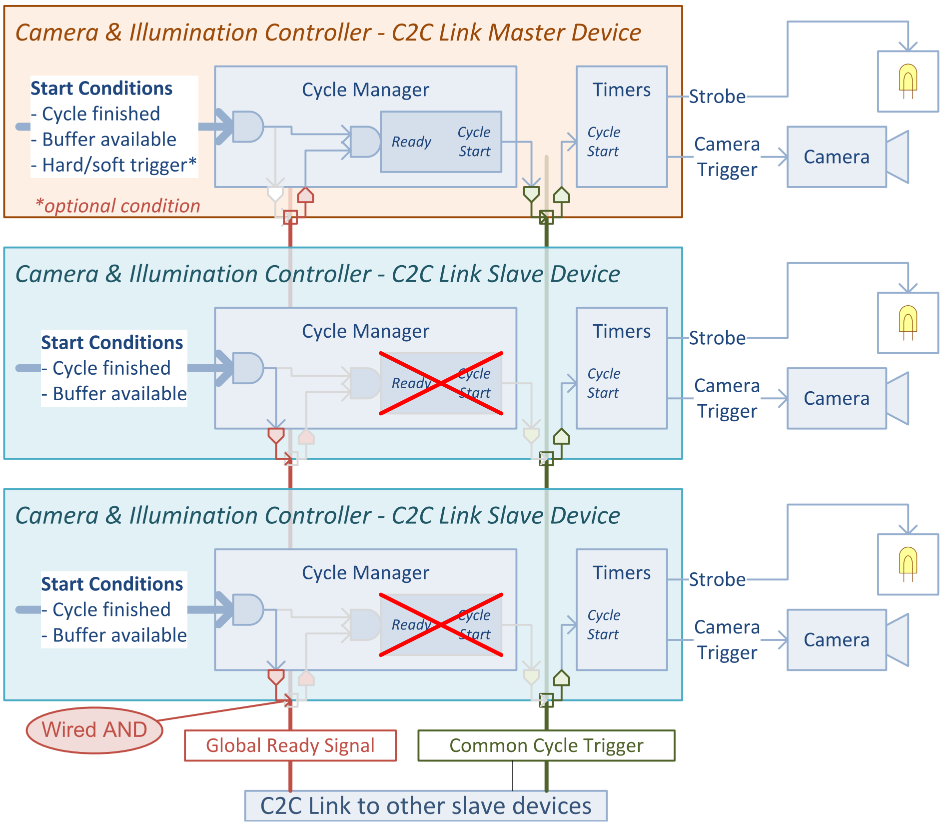
Parameter	Min.	Typ.	Max.	Units
(1) Master IntraPC interconnection	0	5	10	ns
(2) Master **1636 InterPC C2C-Link Adapter** – IntraPC-to-InterPC delay		15		ns
(5) Master to slave1 InterPC LAN cable – 20 m		100		ns
(3) Slave1 **1636 InterPC C2C-Link Adapter** – InterPC-to-IntraPC delay		20		ns
(1) Slave1 IntraPC interconnection	0	5	10	ns
Total Trigger Delay for devices of the Slave1 InterPC segment		145		ns

For devices belonging to the same IntraPC segment as the Slave2 adapter, there are 7 delay elements to consider:

Parameter	Min.	Typ.	Max.	Units
(1) Master IntraPC interconnection	0	5	10	ns
(2) Master **1636 InterPC C2C-Link Adapter** – IntraPC-to-InterPC delay		15		ns
(5) Master to slave1 InterPC LAN cable – 20 m		100		ns
(4) Slave1 **1636 InterPC C2C-Link Adapter** – InterPC-to-InterPC delay		0		ns
(5) Slave1 to slave2 InterPC LAN cable – 20 m		100		ns
(3) Slave2 **1636 InterPC C2C-Link Adapter** – InterPC-to-IntraPC delay		20		ns
(1) Slave2 IntraPC interconnection	0	5	10	ns
Total Trigger Delay for devices of the Slave2 InterPC segment		245		ns

Cycle Trigger Synchronization

Principle



One C2C-Link master and two C2C-Link slave devices (With Global Ready)

Common cycle trigger

The CIC synchronization is achieved by sharing a common Cycle Trigger event between all involved CIC 's using the C2C-Link interconnections. The C2C-Link interconnects two or more devices. One device is named*C2C-Link Master Device*, the others are named *C2C-Link Slave Device*.

The Cycle Trigger is elaborated by the Cycle Manager of the "*C2C-Link Master Device* and broadcast on the C2C-Link via the "C2C-Link Synchronization Tool" on page 155 of the I/O toolbox. All the participating C2C-Link devices (1 Master and one or more slaves) uses the shared C2C1 event stream as event source for the camera cycle trigger.

The CIC Cycles of all participating devices start simultaneously. However, the Cycle Timing parameters (exposure time, strobe pulse width and strobe delay) can be configured individually.

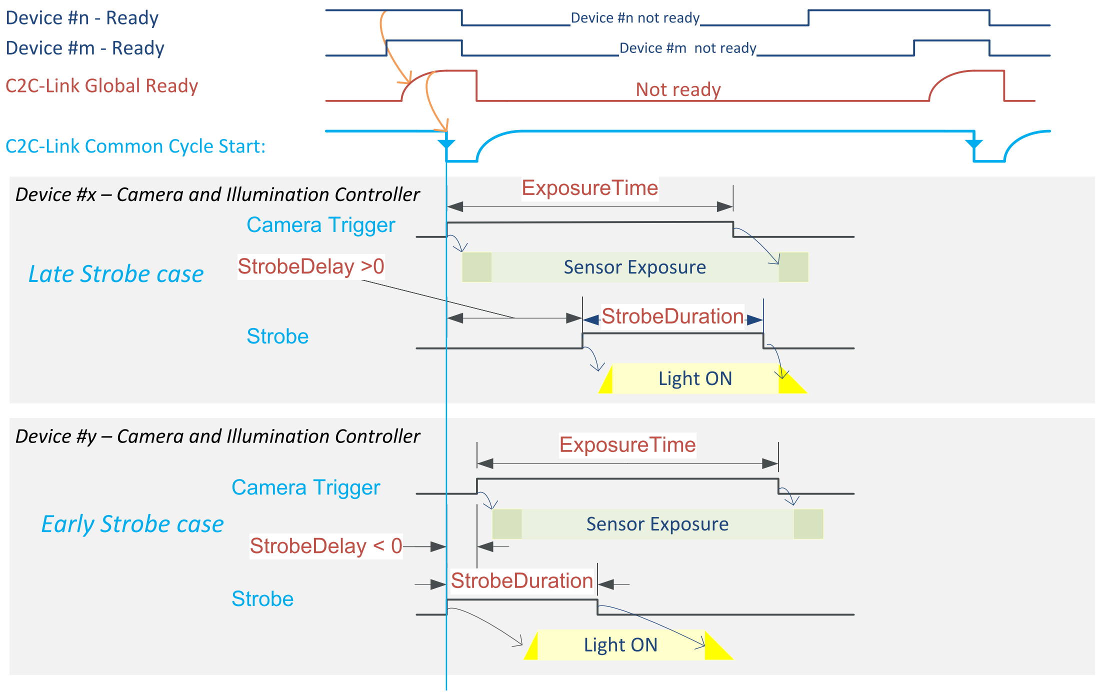
Global Ready signal

Global Ready applies only to area-scan firmware variants!

The Global Ready signal is elaborated by all C2C-Link slaves using a wired-and scheme. A C2C-Link Slave forces the Global Ready signal to 'false' until its start conditions are all satisfied.

The C2C-Link Master device further delays the assertion a Cycle Trigger event while the Global Ready signal is false.

Timing Diagram



CIC Synchronization through C2C-Link timing diagram (With Global Ready)

The above diagram shows the timing diagram of two consecutive common CycleTrigger events with Global Ready :

The C2C-Link Global Ready signal is held low until the Ready of all participating devices is true preventing the C2C-Link Master to issue a start event. When released by all participating devices, it ramps up rapidly with a rise time of maximum 100 ns.

As soon as the C2C-Link Global Ready signal is confirmed to be high, the master device asserts an abrupt going low transition on the Common Cycle Start signal; this edge is propagated to all the Start inputs of the timers of all participating devices.

As soon as the cycle has started, every CIC forces the ready low as long as all the local conditions to initiate the next cycle are not satisfied.

The timers of each device issue a Camera Trigger and a Strobe, with their respective delay and duration settings. Usually, the settings are identical for all participating devices; but the application is allowed to apply different ones, if needed.

The shortest Cycle Start period allowed by the C2C-Link is 400 ns; allowing a theoretical frequency limit of 2.5 MHz.

C2C-Link Setup Procedure

Hardware setup

This step is specific to each C2C-Link Configuration:

IntraCard C2C-Link configuration

This configuration exclusively uses FPGA internal resources to build the C2C-Link interconnect; it doesn't require any additional hardware!

IntraPC C2C-Link configuration

This configuration requires one accessory cable such as the **3303 C2C-Link Ribbon Cable** or a custom-made C2C-Link cable for each PC.

Insert a C2C-Link female connector of the C2C-Link cable into the C2C-Link pin header connector of each participating Coaxlink cards.

InterPC C2C-Link configuration

This configuration requires one **1636 InterPC C2C-Link Adapter** for each PC and one RJ 45 CAT 5 STP straight LAN cable for each adapter but the last one.

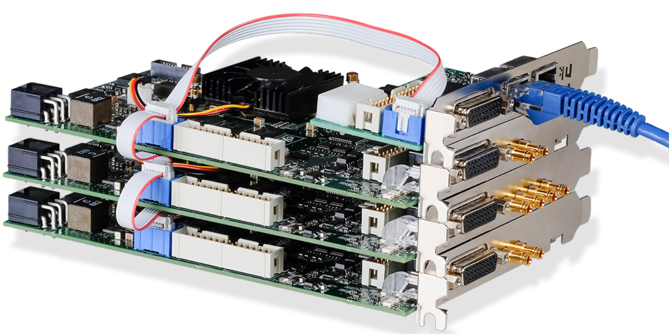
In each participating PC:

1. Install **1636 InterPC C2C-Link Adapter** into a free slot and secure the bracket.
2. Connect the adapter to a power source.

See also: "Adapter Powering" topic in the 1636 section of the hardware manual.

3. Using one **3303 C2C-Link Ribbon Cable**, bind together the C2C-Link connectors of all the participating cards together with the C2C-Link of the adapter card.
4. Using LAN Cables, interconnect the adapters.

See also: "InterPC Interconnect" topic in the 1636 section of the hardware manual.



IntraPC segment of an InterPC configuration

GenAPI setup

Master C2C-Link device

- Assign value Master to C2CLinkConfiguration of the GenTL Device module
- Configure the I/O Toolbox C2C1 tool of the GenTL Interface module to share Cycle Trigger:
  - Assign value CycleTrigger to C2CLinkSynchronizationToolSource
  - Assign value Immediate to C2CLinkSynchronizationToolClock
- Optional, configure the I/O Toolbox C2C2 and C2C3 tools of the GenTL Interface module according to the application requirements

Slave C2C-Link devices

Assign value Slave to C2CLinkConfiguration of the GenTL Device module

11.11. OEM Safety Key

Introducing OEM Safety Key

The OEM Safety Key capability allows the application to:

- Program an "OEM safety key" in the non-volatile memory of the Coaxlink card.
- Retrieve the encrypted version of the OEM safety key just programmed.
- Check a key against the programmed OEM safety key or its encrypted version.

OEM Safety Key

The OEM Safety Key is an application-defined string of characters. Any character except the null character is allowed. The string length is unlimited.

Key Programming

When the application sets the ProgramOemSafetyKey GenICam feature with the OEM Safety Key value, the Coaxlink driver computes an encrypted version of the OEM Safety Key and stores it in the non-volatile memory of the Coaxlink card.

The encrypted value can be retrieved by getting the value of EncryptedOemSafetyKey immediately after having set ProgramOemSafetyKey.

> **WARNING**
> Only the same application process having set ProgramOemSafetyKey is allowed to retrieve the encrypted value. This is only allowed until any other GenICam feature is set.

Key Checking

The application has to select one OemSafetyKeyVerification value of the

In order to verify the OEM Safety Key of a Coaxlink card, the application sets a "challenge" value to the CheckOemSafetyKey[selector] feature.

When the [selector] argument is set to EncryptedKey, the set action terminates normally only when the challenge string is identical to the encrypted OEM Safety Key string.

When the [selector] argument is set to ProgrammingKey, the set action terminates normally only when the challenge string is identical to the programming OEM Safety Key string.

When the [selector] argument is set to ProgrammingKeyOrEncryptedKey, or omitted, the set action terminates normally only when the challenge string is identical to the original OEM Safety Key string or to the encrypted OEM Safety Key string.

Euresys recommends using the EncryptedKey selector. This improves the security level since the programming key doesn't need to appear anywhere in the end user application. Having only the encrypted key, the end user cannot retrieve the original programming key.

Using OEMSafetyKey

Programming Step – Option A

Using GenICam Browser:

- Go to the GenApi tab of the interface module.
- Write a secret key to ProgramOemSafetyKey
- Copy the value of EncryptedOemSafetyKey and paste it somewhere appropriate.

NOTE
There is a direct relationship between the *programming key* and the *encrypted key*. A given *programming key* will always lead to the same *encrypted key*, even on different computers or with different Coaxlink cards. This makes it possible to read the *encrypted key* once and hard-code this value in the application that must be protected by the OEM safety key.

Programming Step – Option B

Using a custom application:

1. Program the OEM safety key of the Coaxlink card by writing a secret key to ProgramOemSafetyKey.
2. Read back the encrypted key by reading EncryptedOemSafetyKey. Write this value somewhere appropriate.

```
grabber.set<InterfacePort>("ProgramOemSafetyKey","plain-text key");          // 1
std::stringencryptedKey=grabber.get<InterfacePort>("EncryptedOemSafetyKey");       // 2
```

Verification Step

In the application that must be protected by the OEM key:

```
InterfacePort>("CheckOemSafetyKey[EncryptedKey]", "encrypted key retrieved in the programming
step");
```

NOTE
Even if the *encrypted key* is discovered and an attacker uses it to reprogram cards, the above verification will fail.