

# Coaxlink

Coaxlink 10.3.1



PCI  
EXPRESS<sup>TM</sup>

CoaxPress

### *Terms of Use*

EURESYS s.a. shall retain all property rights, title and interest of the documentation of the hardware and the software, and of the trademarks of EURESYS s.a.

All the names of companies and products mentioned in the documentation may be the trademarks of their respective owners.

The licensing, use, leasing, loaning, translation, reproduction, copying or modification of the hardware or the software, brands or documentation of EURESYS s.a. contained in this book, is not allowed without prior notice.

EURESYS s.a. may modify the product specification or change the information given in this documentation at any time, at its discretion, and without prior notice.

EURESYS s.a. shall not be liable for any loss of or damage to revenues, profits, goodwill, data, information systems or other special, incidental, indirect, consequential or punitive damages of any kind arising in connection with the use of the hardware or the software of EURESYS s.a. or resulting of omissions or errors in this documentation.

This documentation is provided with Coaxlink 10.3.1 (doc build 2052).  
© 2018 EURESYS s.a.

# Contents

1. About This Document .....	7
1.1. Document Scope .....	7
1.2. Document Revision History .....	8
1.3. Document Changes .....	10
2. Coaxlink Architecture .....	11
2.1. Main Elements .....	12
2.2. Block Diagrams .....	14
3. CoaXPress Host Interface .....	18
3.1. Host Interface Specification .....	19
3.2. Firmware Variants per Product .....	23
3.3. Camera to Host Connections Maps .....	29
3.4. Link Configuration .....	34
3.5. CoaXPress 1.1.1 Discovery .....	35
3.6. Power Over CoaXPress .....	36
3.7. CoaXPress I/O Channel .....	39
3.8. CoaXPress Host To Device Trigger .....	40
3.9. CoaXPress Lamps .....	46
3.10. Connection Test .....	47
3.11. CoaXPress Link Validation Tool .....	48
4. Image Data Path .....	54
4.1. FIFO Buffer .....	55
4.2. Acquisition Gate .....	56
4.3. Pixel Data Processing .....	57
4.4. Pixel Data Processing Configurations .....	59
4.5. Pixel Component Unpacking .....	62
4.6. Pixel Component Re-Ordering .....	64
4.7. Endianness Conversion .....	65
4.8. Pixel Ordering .....	66
4.9. Image Data Transfer .....	67

5. Camera Control Principles .....	71
5.1. Camera Cycle .....	72
5.2. Camera Cycle Concatenation Rules .....	74
5.3. Camera Control Methods .....	76
6. Illumination Control Principles .....	78
6.1. Illumination Devices .....	79
6.2. Aligning Camera and Illumination Cycles .....	80
7. Camera and Illumination Controller .....	82
7.1. CIC Block Diagram .....	82
7.2. Cycle Timing Machine .....	83
7.3. Cycle Manager .....	85
7.4. Cycle Trigger Manager .....	86
7.5. Sequence Manager .....	88
7.6. CIC Output Signals Routing .....	89
7.7. CIC Timing Diagrams .....	89
Cycle Timing Diagrams .....	90
Consecutive Overlapping CIC Cycles .....	92
Cycle Sequence Timing Diagrams .....	97
8. General Purpose I/O .....	99
8.1. I/O Line Types .....	100
8.2. Available I/O Lines .....	101
8.3. I/O Lines Usage .....	103
8.4. I/O Control Blocks .....	104
8.5. Line Mode Control .....	106
8.6. I/O Lines Specifics .....	107
8.7. Line Polarity Control .....	108
8.8. Filter Control .....	109
8.9. Line Source Selection .....	110
8.10. Logical I/O Line State .....	111
8.11. Physical I/O Line State .....	112
8.12. Line Driver Physical Output States .....	113
8.13. Initial States .....	114
9. I/O Toolbox .....	115
9.1. Introducing the I/O Toolbox .....	116

9.2. I/O Toolbox Composition .....	119
9.3. Line Input Tool .....	121
9.4. Quadrature Decoder Tool .....	122
9.5. Event Input Tool .....	125
9.6. Divider Tool .....	126
9.7. Multiplier/Divider Tool .....	127
9.8. Delay Tool .....	130
9.9. User Actions Tool & User Output Register .....	132
10. Event Signaling And Counting .....	136
10.1. Introduction .....	136
10.2. Custom Events Sources .....	139
10.3. Event Specific Context Data .....	142
10.4. About GenTL Signaling .....	144
11. Advanced Features .....	145
11.1. Sub-link Acquisition .....	146
11.2. Multi-Stream Acquisition .....	148
11.3. CoaXPress Data Forwarding .....	149
Data Forwarding Principles .....	150
Data Forwarding Connection Schemes .....	152
Line-scan Triggers Synchronization .....	153
Configuration Script Example .....	154
11.4. Flat Field Correction .....	155
What is Flat Field Correction? .....	156
Coaxlink FFC .....	158
FFC Wizard Sample Program .....	161
11.5. Lookup Table Processing .....	164
Introduction to LUT Processing .....	164
LUT Processing Availability .....	165
Monochrome Lookup Table Processing .....	167
LUT Content Definition .....	168
LUT Setup Procedure .....	175
11.6. Bayer CFA Decoding .....	178
Bayer CFA to RGB Conversion .....	179
Using Bayer CFA Decoder .....	183
11.7. Laser Line Extraction .....	186
Introduction .....	187
Laser Line Extraction Algorithms .....	188
LLE Processing Core Implementation .....	189
LLE Processing Core Characteristics .....	190
Linear Filter .....	192
Coring Threshold .....	193

Maximum Detection .....	194
Peak Detection .....	195
Center of Gravity .....	196
Use Case Example .....	197
Objective .....	198
Defining the ROI .....	199
Setting Filtering and Thresholding .....	201
Defining the LLE Algorithm .....	202
Defining Scan Length and Buffer Size .....	204
Appendix .....	205
<b>11.8. Line Scan Acquisition .....</b>	<b>206</b>
Line Scan Acquisition Principles .....	207
Line Scan Acquisition Use cases .....	212
<b>11.9. CIC Synchronization .....</b>	<b>216</b>
CIC Synchronization Principle .....	217
CIC Synchronization Timing Diagram .....	219
<b>11.10. C2C-Link .....</b>	<b>220</b>
C2C-Link Description .....	221
C2C-Link Specification .....	223
Trigger Propagation Delays .....	225
C2C-Link Use Cases .....	227
C2C-Link Availability .....	228
C2C-Link Setup .....	229
3303 C2C-Link Ribbon Cable .....	230
Custom C2C-Link Ribbon Cable Assembly .....	231
1636 InterPC C2C-Link Adapter .....	232
Product Pictures .....	233
Hardware Description .....	234
Using 1636 as HD26F I/O Adapter .....	236
Using 1636 as C2C-Link Extender .....	237
Adapter Powering .....	237
InterPC Interconnect .....	237
Lamps .....	238
<b>11.11. OEM Safety Key .....</b>	<b>241</b>
Introducing OEM Safety Key .....	241
Using OEMSafetyKey .....	242

# 1. About This Document

1.1. Document Scope .....	7
1.2. Document Revision History .....	8
1.3. Document Changes .....	10

## 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 10.3.1

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 Revision History

Date	Version	Description
2016-08-05	6.0	Functional Guide of Coaxlink driver 6.0 <ul style="list-style-type: none"> <li>• Add 1634 Coaxlink Duo PCIe/104 product</li> <li>• Add "User Actions Tool &amp; User Output Register" on page 132</li> </ul>
2016-09-08	6.1	Functional Guide of Coaxlink driver 6.1 <ul style="list-style-type: none"> <li>• Extend IntraPC C2C-Link limit to 0.6 m</li> <li>• Add 1636 InterPC C2C-Link Adapter</li> <li>• Extend GPIO Line sources to 'Device&lt;0:3&gt; Stream0 Start Of Camera Readout'</li> <li>• Extend DIV tool range to 16-bit</li> <li>• Improve security level of OEM Safety Key</li> <li>• Preliminary support of sub-link acquisition for a particular 8-connection camera</li> </ul>
2016-09-29	6.1.1	Functional Guide of Coaxlink driver 6.1.1
2016-11-24	6.2.2	Functional Guide of Coaxlink driver 6.2.2 <ul style="list-style-type: none"> <li>• Add support of lookup tables for monochrome pixels</li> <li>• Add data stream line-padding and stripe-padding</li> </ul>
2017-02-06	6.2.4	Functional Guide of Coaxlink driver 6.2.4
2017-05-29	8.0	Functional Guide of Coaxlink driver 8.0 <ul style="list-style-type: none"> <li>• Add support of 1638 Coaxlink Quad CXP-3</li> </ul>
2017-06-20	9.0	Functional Guide of Coaxlink driver 9.0 <ul style="list-style-type: none"> <li>• Add support of 1637 Coaxlink Quad 3D-LLE</li> <li>• Extend the I/O Toolbox of the 4-camera firmware variant of 1633 Coaxlink Quad G3</li> </ul>
2017-07-03	9.1	Functional Guide of Coaxlink driver 9.1
2017-08-08	9.2	Functional Guide of Coaxlink driver 9.2
2017-11-08	9.3	Functional Guide of Coaxlink driver 9.3
2017-11-22	9.4	Functional Guide of Coaxlink driver 9.4
2018-03-28	9.5	Functional Guide of Coaxlink driver 9.5.2



Date	Version	Description
2018-05-29	10.0	Functional Guide of Coaxlink driver 9.5.2 <ul style="list-style-type: none"><li>• Add support of macOS</li><li>• Add support of 3602 Coaxlink Octo</li></ul>
2018-06-29	10.1	Functional Guide of Coaxlink driver 10.1
2018-08-14	10.1.2	Functional Guide of Coaxlink driver 10.1.2
2018-09-19	10.2.1	Functional Guide of Coaxlink driver 10.2.1
2018-10-24	10.3	Functional Guide of Coaxlink driver 10.3.0

## 1.3. Document Changes

### Coaxlink 10.3

---

The following topics were added:

- "Width Increment Step " on page 23
- "Flat Field Correction" on page 155
  - "What is Flat Field Correction?" on page 156
  - "Coaxlink FFC" on page 158
  - "FFC Wizard Sample Program " on page 161

The following topics were revised:

- "Pixel Data Processing" on page 57
- "Pixel Data Processing Configurations" on page 59
- "Delay Tool" on page 130

The following sections were moved to "Advanced Features" on page 145:

- "CoaXPress Data Forwarding" on page 149
- "CIC Synchronization" on page 216
- "C2C-Link" on page 220
- "Line Scan Acquisition" on page 206
- "Laser Line Extraction" on page 186
- "OEM Safety Key" on page 241

## 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 99 block gathers all the I/O ports of the card.

#### *I/O Toolbox*

The "I/O Toolbox" on page 115 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 56 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 55 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 57 performs on-the-fly pixel processing.

### Data Formatting

"Pixel Component Unpacking" on page 62, "Pixel Component Re-Ordering" on page 64, "Endianness Conversion" on page 65 and "Pixel Ordering" on page 66 operations are configured according the desired pixel output format.

### Image Data Transfer

The "Image Data Transfer" on page 67 is transfers the image data to the destination buffer.

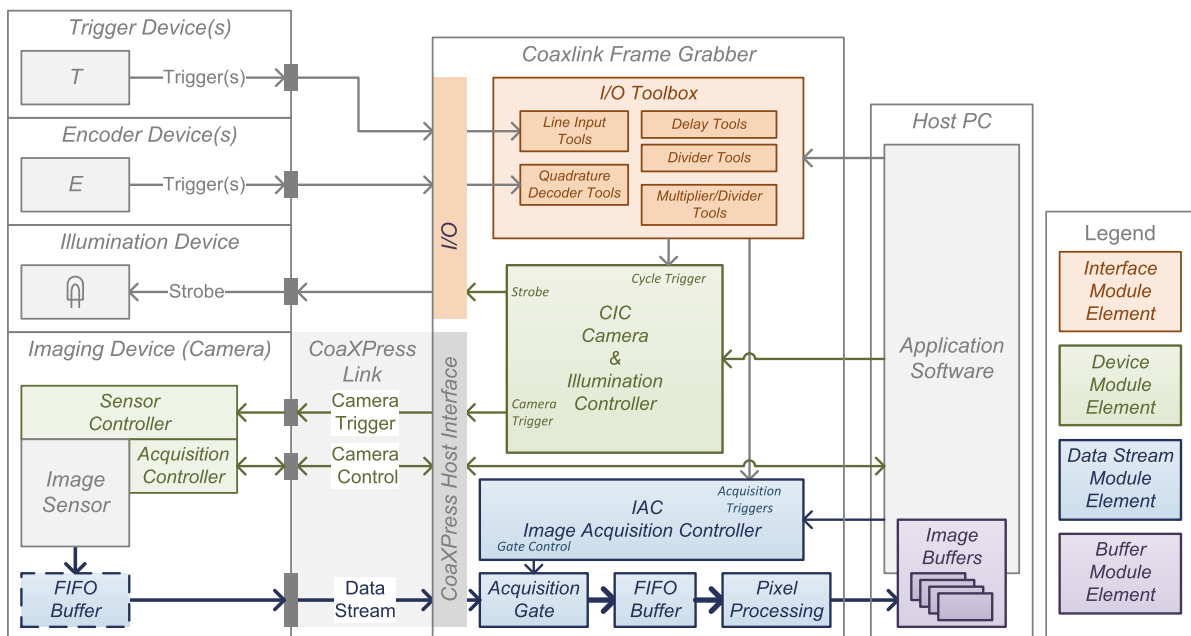
## 2.2. Block Diagrams

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

Applies to: Mono Duo Quad QuadG3 QuadG3DF Duo104EMB Duo104MIL  
QuadCXP3 Quad3DLLE Octo QuadCXP12

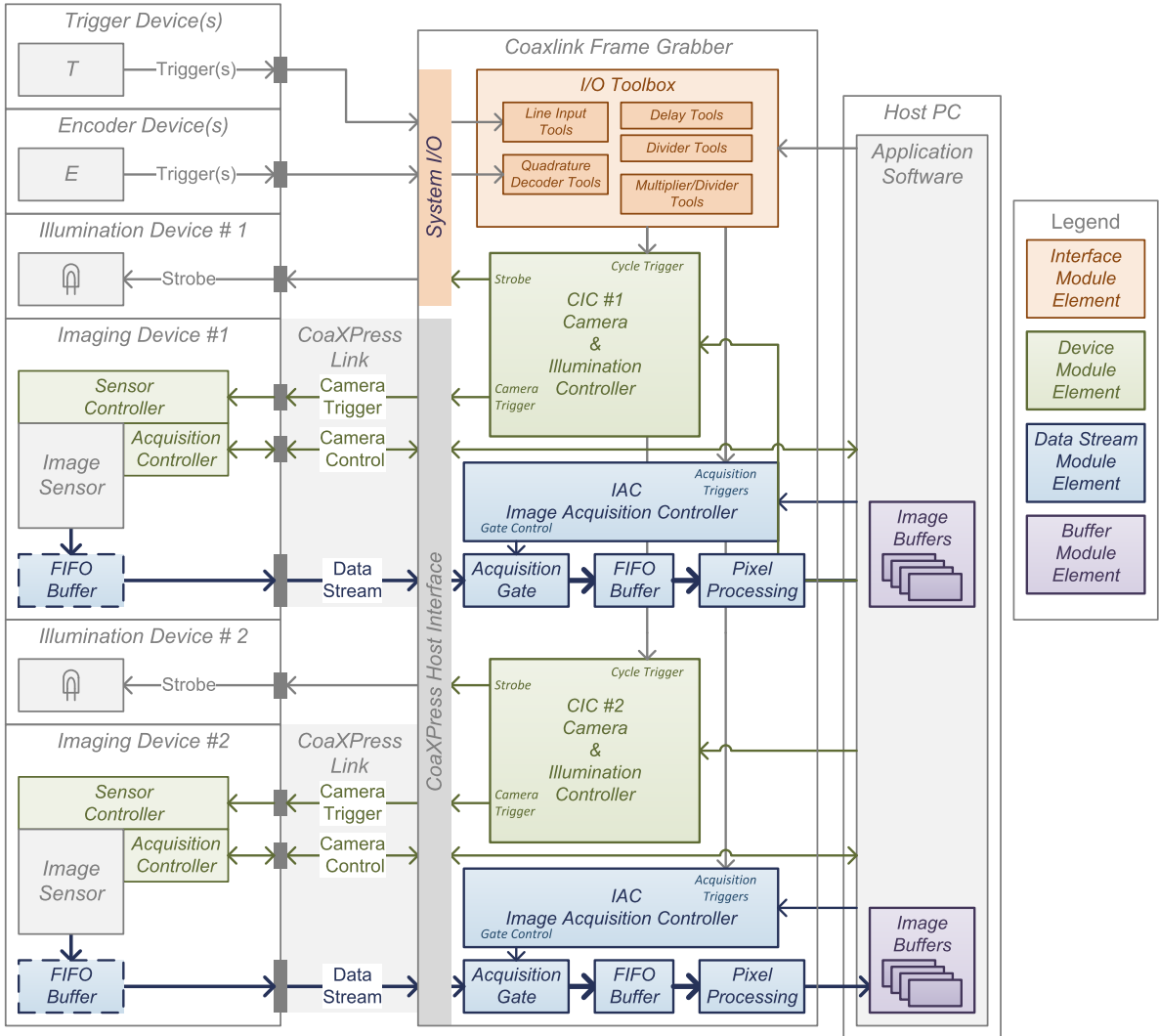


**1-camera, 1-data-stream Coaxlink image acquisition system**

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

## 2-camera, 1-data-stream

Applies to: Duo Quad QuadG3 Duo104EMB Duo104MIL QuadCXP3 Octo  
QuadCXP12

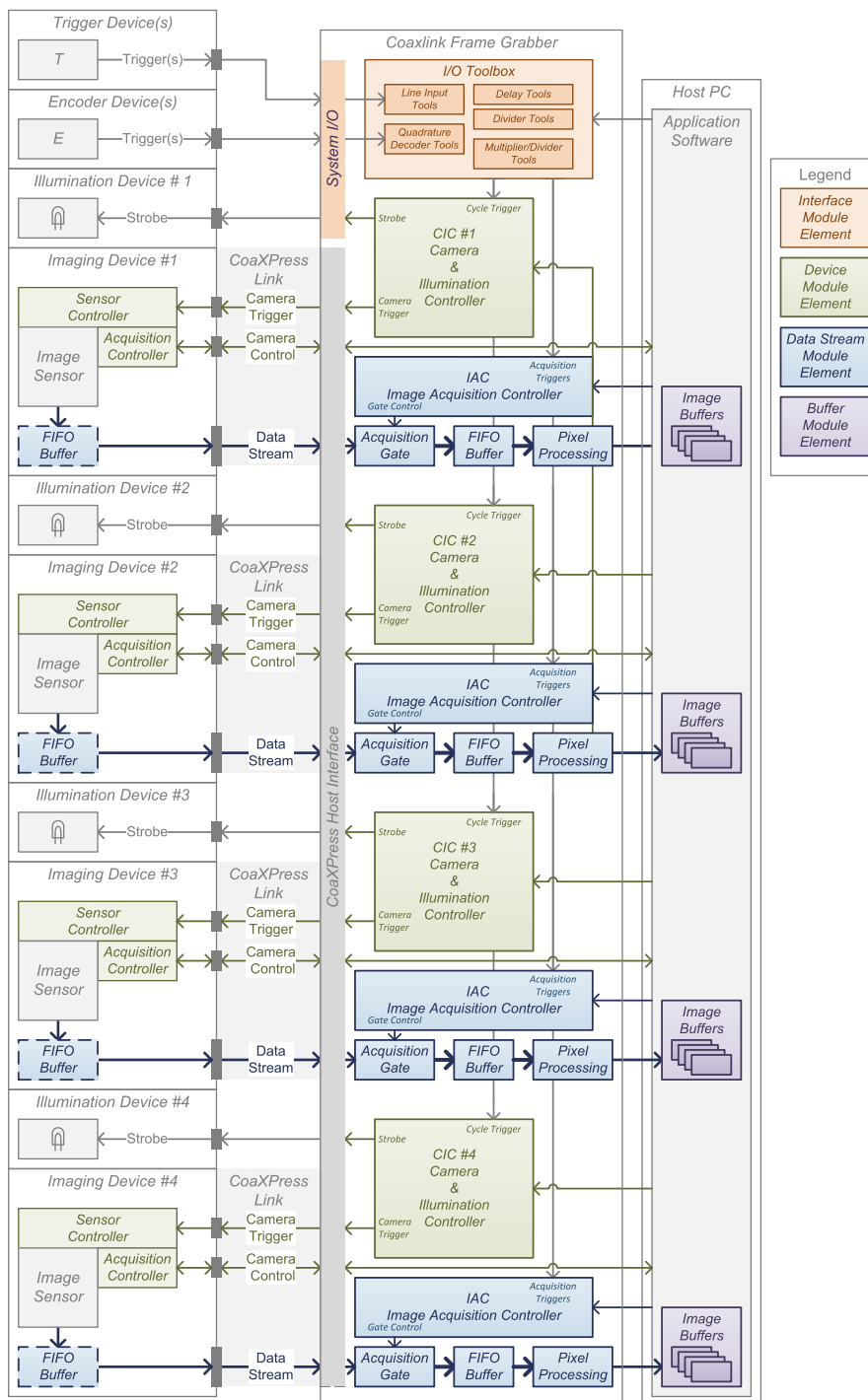


2-camera, 1-data-stream Coaxlink image acquisition system

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

## 4-camera, 1-data-stream

Applies to: QuadG3



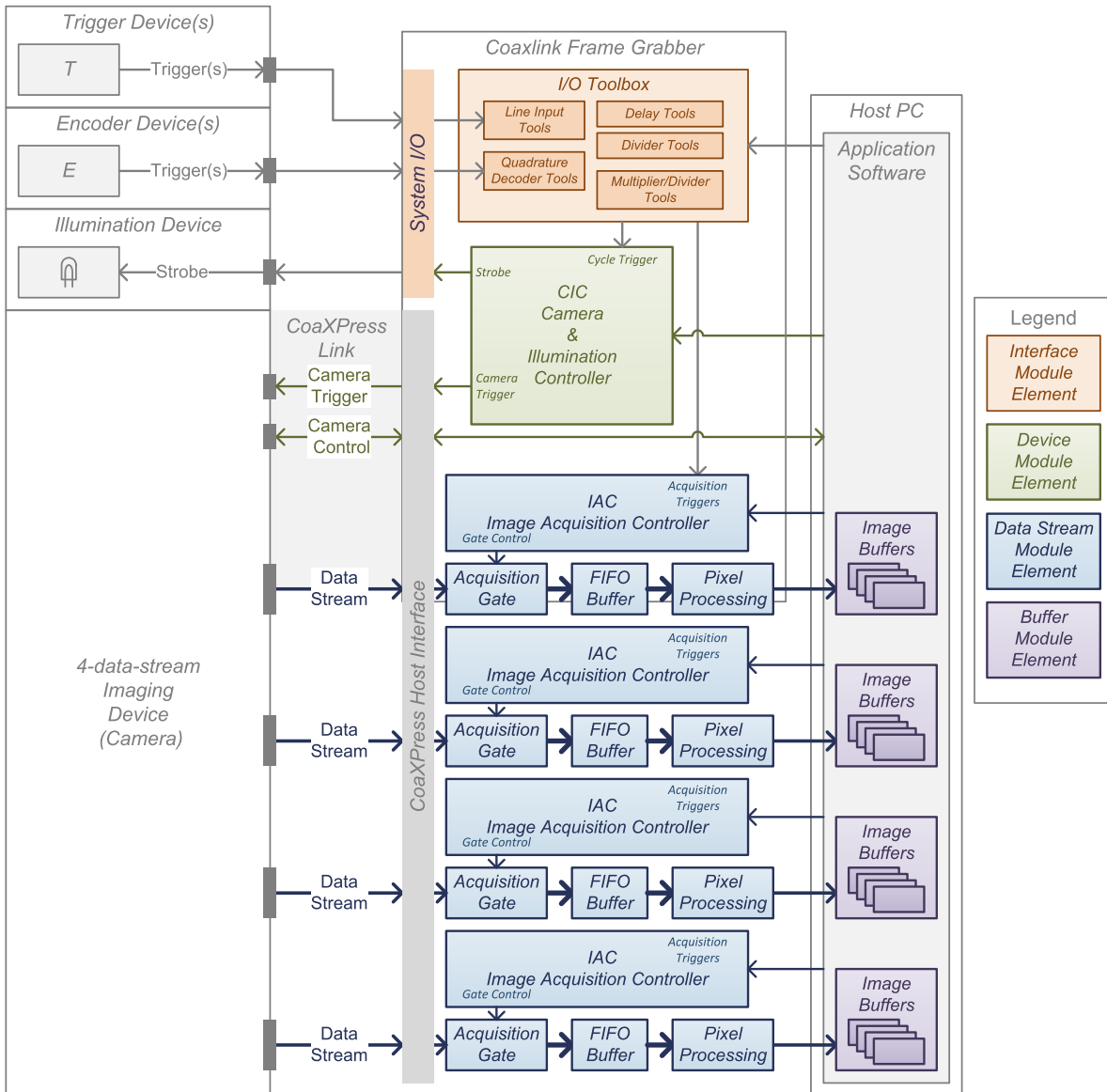
**4-camera, 1-data-stream Coaxlink image acquisition system**

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



## 1-camera, 4-data-stream

Applies to: **QuadG3**



**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-3 DIN 4	QuadCXP3
CXP-6 DIN 1	Mono
CXP-6 DIN 2	Duo Duo104EMB Duo104MIL
CXP-6 DIN 4	Quad QuadG3 QuadG3DF
CXP-6 DIN 8	Octo
CXP-12 HD-BNC 4	QuadCXP12

Down-connection speed	Applicable products
1.25, 2.5, and 3.125 GT/s	QuadCXP3
1.25, 2.5, 3.125, 5.0, and 6.25 GT/s	All products but QuadCXP3
1.25, 2.5, 3.125, 5.0, 6.25, 10.0 and 12.5 GT/s	QuadCXP12

Maximum aggregated input data rate	Applicable products
625,000 Bytes/s	Mono
1,250,000 Bytes/s	Duo Duo104EMB Duo104MIL QuadCXP3
2,500,000 Bytes/s	Quad QuadG3 QuadG3DF
5,000,000 Bytes/s	Octo QuadCXP12




Maximum deliverable PoCXP power	Applicable products
17 W per connector	All products

Devices count	Applicable firmware variants
1	
1 or 2	
1 up to 4	
1 up to 8	

## 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	
Area-scan camera, up to 4 data streams	
Line-scan camera, 1 data stream	

The above table excludes the special camera types used in "CoaXPress Data Forwarding" on page 149 and "Sub-link Acquisition" on page 146 configurations.

Connections count (per camera)	Applicable products/ firmware variants
1	Refer to "Firmware Variants per Product" on page 23 in the release notes.
1 or 2	
1, 2 or 4	
1, 2, 4 or 8	
8	Refer to "Sub-link Acquisition" on page 146 configurations

## Data Stream Specifications

Specifications having the scope of oneGenTL Data Stream module

Maximum stream packet size	Applicable products
8,192 Bytes	QuadCXP3
16,384 Bytes	All products but QuadCXP3
Image stream format	Applicable firmware variants
Rectangular Image (YSize > 0)	
Rectangular Image (YSize = 0)	
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	Not supported
1X-2YE	Not supported

**Note: 1X-1Y2** is used in dual-tap cameras having a single zone in the vertical direction and delivering simultaneously 2 consecutive lines

**Note: 1X-2YE** is used in dual-tap cameras having two zones in the vertical direction and delivering simultaneously 2 lines starting from the top and the bottom lines of the image.

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	All firmware variants

Pixel format	Applicable firmware variants
BayerGR16, BayerRG16, BayerGB16, BayerBG16	
RGB8, RGB10, RGB12, RGB14, RGB16	All firmware variants
RGBA8, RGBA10, RGBA12, RGBA14, RGBA16	All firmware variants

### Width Increment Step

**Important:** 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 QuadCXP3 Quad3DLLE

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

Applies to: QuadG3 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

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

## 3.2. Firmware Variants per Product

### 1630 Coaxlink Mono

Firmware Variant	Description	Host Connections Map	Stream Buffer Size	Advanced Processing
1-camera	One 1-connection area-scan camera	1D1	512 MB	LUT

### 1631 Coaxlink Duo

Firmware Variant	Description	Host Connections Map	Stream Buffer Size	Advanced Processing
1-camera	One 1- or 2-connection area-scan camera	1D2	1 GB	LUT
2-camera	One or two 1-connection area-scan cameras	2D11	512 MB	LUT
1-camera, line-scan	One 1- or 2-connection line-scan camera	1D2	1 GB	LUT
2-camera, line-scan	One or two 1-connection line-scan cameras	2D11	512 MB	LUT

### 1632 Coaxlink Quad

Firmware Variant	Description	Host Connections Map	Stream Buffer Size	Advanced Processing
1-camera	One 1- or 2- or 4-connection area-scan camera	1D4	1 GB	LUT
2-camera	One or two 1- or 2-connection area-scan cameras	2D22	512 MB	LUT
1-camera, line-scan	One 1- or 2- or 4-connection line-scan camera	1D4	1 GB	LUT



### 1633 Coaxlink Quad G3

Firmware Variant	Description	Host Connections Map	Stream Buffer Size	Advanced Processing
1-camera	One 1- or 2- or 4-connection area-scan camera, 1 data-stream	1D4	1 GB	LUT CFA
			512 MB	FFC LUT CFA
1-camera, 4-data-stream	One 1- or 2- or 4-connection area-scan camera, up to 4 data streams	1D4S4	256 MB	-
2-camera	One or two 1- or 2-connection area-scan cameras	2D22	512 MB	LUT
4-camera	One or two or three or four 1-connection area-scan cameras	4D1	256 MB	LUT
1-camera, line-scan	One 1- or 2- or 4-connection line-scan camera	1D4	1 GB	LUT
2-camera, line-scan	One or two 1- or 2-connection line-scan cameras	2D22	512 MB	LUT
4-camera, line-scan	One or two or three or four 1-connection line-scan cameras	4D1111	256 MB	LUT
1-slm-camera	Master 4-connection sub-link of an 8-connection area-scan camera	1D8SL4	1 GB	LUT
1-sls-camera	Slave 4-connection sub-link of an 8-connection area-scan camera	1D8SL4	1 GB	LUT

### 1629 Coaxlink Duo PCIe/104-EMB, 1634 Coaxlink Duo PCIe/104-MIL

Firmware Variant	Description	Host Connections Map	Stream Buffer Size	Advanced Processing
1-camera	One 1- or 2-connection area-scan camera	1D2	512 MB	LUT
2-camera	One or two 1-connection area-scan cameras	2D11	256 MB	LUT
1-camera, line-scan	One 1- or 2-connection line-scan camera	1D2	256 MB	LUT

### 1635 Coaxlink Quad G3 DF

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

### 1637 Coaxlink Quad 3D-LLE

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

### 1638 Coaxlink Quad CXP-3

Firmware Variant	Description	Host Connections Map	Stream Buffer Size	Advanced Processing
4-camera	One or two or three or four 1-connection area-scan cameras	4D1111	128 MB	-

### 3602 Coaxlink Octo

Firmware Variant	Description	Host Connections Map	Stream Buffer Size	Advanced Processing
1-camera	One 1- or 2- or 4- or 8-connection area-scan camera	1D8	2 GB	LUT CFA
2-camera	One or two 1- or 2- or 4-connection area-scan cameras	2D44	1 GB	LUT CFA
			512 MB	FFC LUT CFA
4-camera	One or two or three or four 1- or 2-connection area-scan cameras	4D2222	512 MB	LUT
5-camera	One 1- or 2- or 4-connection area-scan camera and one or two or three or four 1-connection area-scan cameras	5D41111	1GB 256 MB	LUT
8-camera	Up to eight 1-connection area-scan cameras	8D11111111	256 MB	LUT
1-camera, line-scan	One 1- or 2- or 4- or 8-connection line-scan camera	1D8	2 GB	LUT
2-camera, line-scan	One or two 1- or 2- or 4-connection line-scan cameras	2D44	1 GB	LUT

## 3603 Coaxlink Quad CXP-12

**Note:** Release 10.3.1 provides only the 1-camera firmware variant of 3603 Coaxlink Quad CXP-12. Please contact us if you need one of the missing firmware variants.

Firmware Variant	Description	Host Connections Map	Stream Buffer Size	Advanced Processing
1-camera	One 1- or 2- or 4-connection area-scan camera, 1 data-stream	1D4	2 GB	LUT CFA
2-camera	One or two 1- or 2-connection area-scan cameras	2D22	1 GB	LUT CFA
4-camera	One or two or three or four 1-connection area-scan cameras	4D1	512 MB	LUT
1-camera, line-scan	One 1- or 2- or 4-connection line-scan camera	1D4	2 GB	LUT
2-camera, line-scan	One or two 1- or 2-connection line-scan cameras	2D22	1 GB	LUT
4-camera, line-scan	One or two or three or four 1-connection line-scan cameras	4D1111	512 MB	LUT

## 3.3. Camera to Host Connections Maps

The Host Interface of Coaxlink requires a specific assignment of the Device connections to the Host connectors. Such assignment is named **Host Connection Map**.

The Host Connection Map is hard-coded in the product/firmware variant. The Coaxlink product and firmware variant must be selected according to the required mapping!

### Host Connection Map Naming Convention

---

The **Host Connection 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. *This field is repeated once for each device.*
  - 1 for a single-connection device
  - 2 for 2-connection 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.
  - *This field is omitted when there are no sub-links.*
  - SL4 for a 4-connection 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.

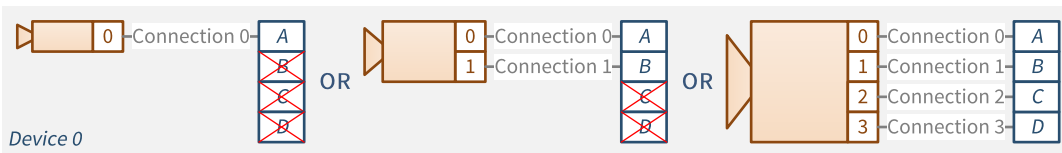
**Standard CoaXPress Devices - One-camera Host Connection Maps (1D1, 1D2, 1D4, 1D8)**



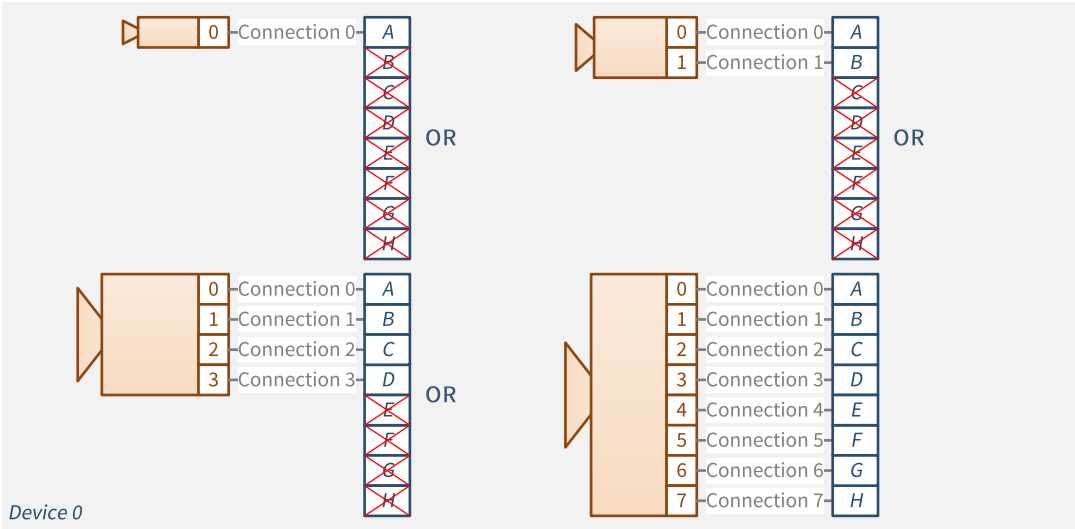
**1D1 host connection map**



**1D2 host connection map**



**1D4 host connection map**

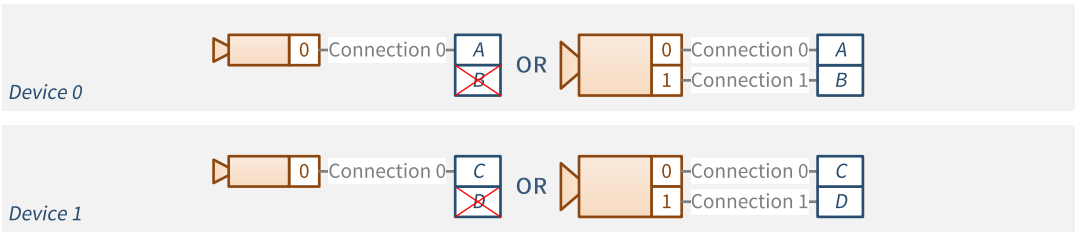


1D8 host connection map

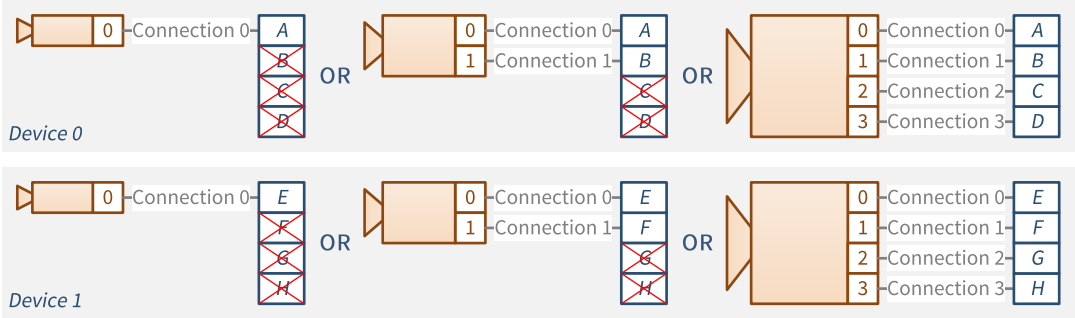
**Standard CoaXPress Devices - Two-camera Host Connection Maps (2D11, 2D22, 2D44)**



2D11 host connection map

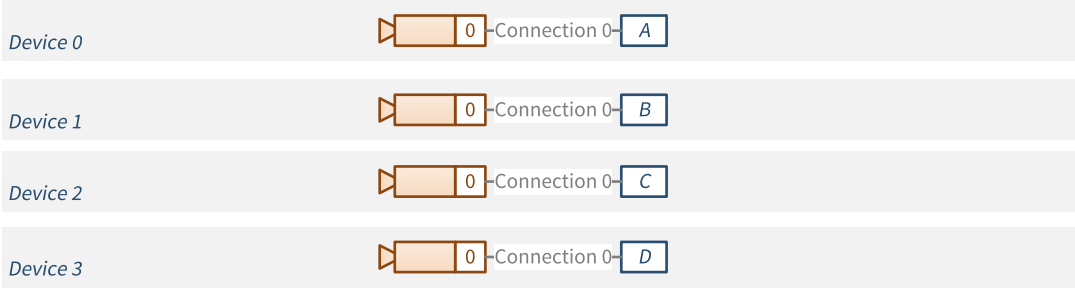


2D22 host connection map

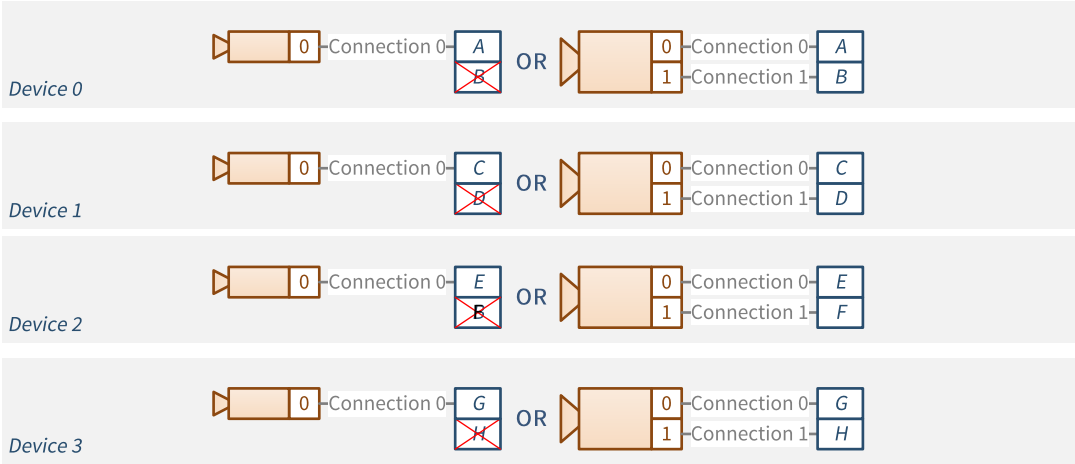


2D44 host connection map

**Standard CoaXPress Devices - Four-camera Host Connection Maps (4D1111, 4D2222)**



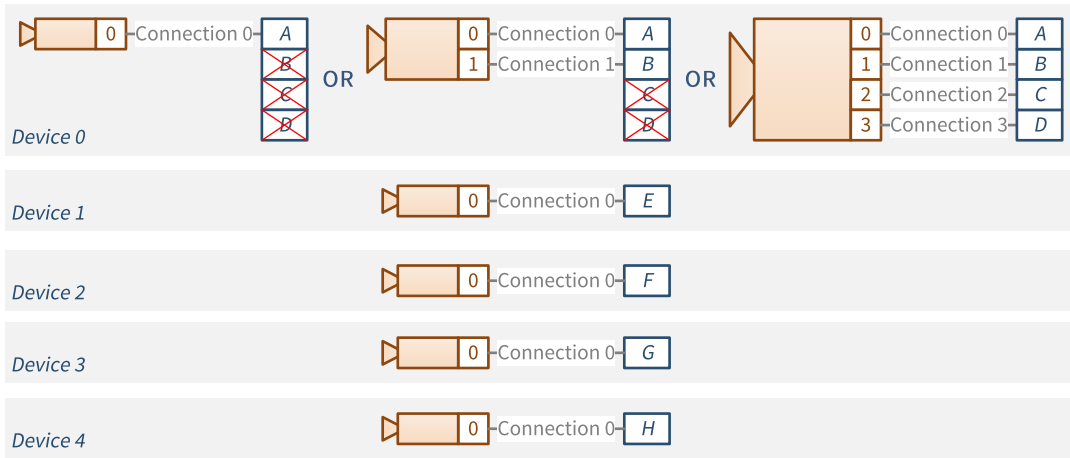
4D1111 host connection map



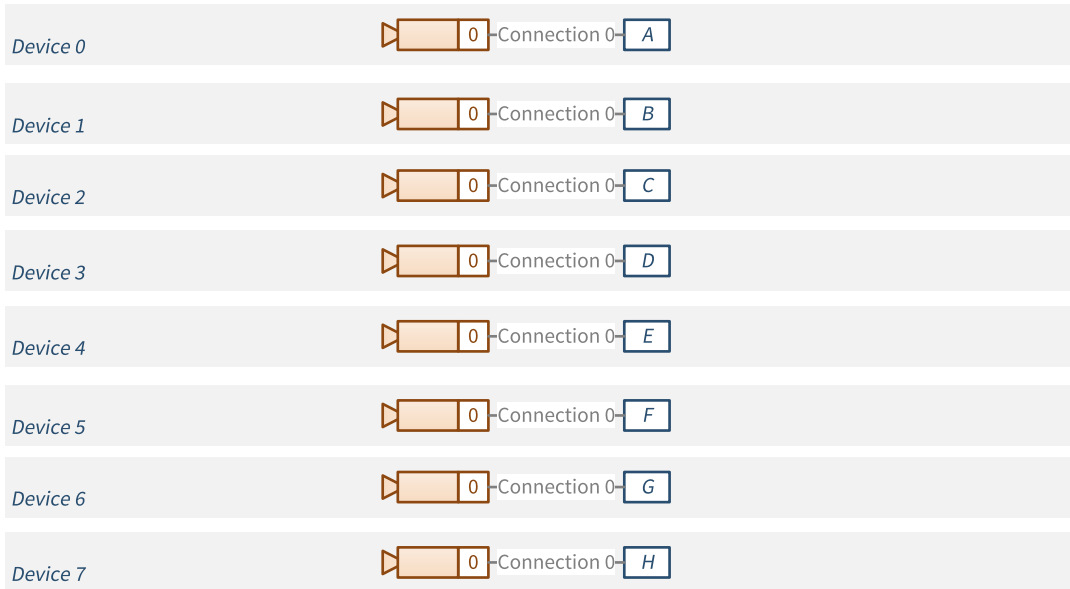
4D2222 host connection map



## Standard CoaXPress Devices - Five- and Eight-camera Host Connection Maps (5D41111, 8D1111111)



### 5D41111 host connection map



### 8D11111111 host connection map

## Host Connection Maps - Non-standard devices

Refer to "[CoaXPress Data Forwarding](#)" on page 149 for the connection schemes of slave Data Forwarding devices.

Refer to "[Sub-link Acquisition](#)" on page 146 for the connection scheme of an 8-connection camera to two 1633 Coaxlink Quad G3.

## 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. CoaXPress 1.1.1 Discovery

The **10.1.3 Discover Devices and Connection Topology** paragraph of the CoaXPress 1.1.1 standard claims:

*"The Host shall read the `ConnectionConfigDefault` register to find the number of expected connections. It shall then write to the `ConnectionConfig` register to enable the number of connections read from `ConnectionConfigDefault`. However it shall not change from the discovery rate at this stage."*

The **10.3.33 ConnectionConfig** paragraph of the CoaXPress 1.1.1 standard claims:

*"This register shall hold a valid combination of the Device connection speed and number of active downconnections. Writing to this register shall set the connection speeds on the specified connections, and the high speed upconnection, if supported. If the new `ConnectionConfig` value results in a change of connection speed, the Device shall acknowledge the `ConnectionConfig` access at the original connection speed. Therefore it shall acknowledge the access before changing connection speed."*

**Note:** *Not all theoretical combinations of connection speed and number of connections may be usable. One register is used to ensure that the two variables are set simultaneously. The XML file and product documentation give valid combinations for the Device. A connection reset sets the value corresponding to the selected discovery rate and one connection.*

Considering that:

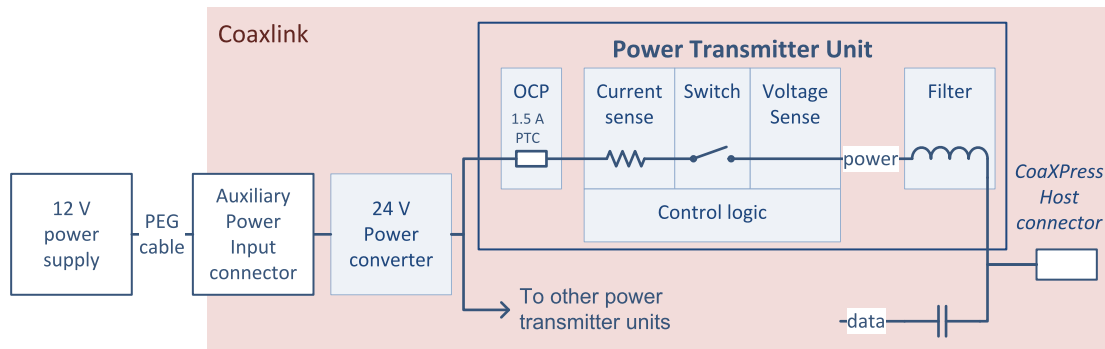
1. the above paragraphs disagree on the value that should be written to the `ConnectionConfig` register,
2. changing the behavior to respect CoaXPress 1.1.1 statements causes issues with some cameras,

the CoaXPress discovery procedure of the Coaxlink driver is not modified to comply with CoaXPress 1.1.1.

At the end of the discovery procedure, the Coaxlink driver sets the speed and the number of the connections of the CoaXPress Link according to the settings of `ConnectionConfigDefault` register of the camera.

## 3.6. Power Over CoaXPress

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



**PoCXP Functional Block Diagram for cards with 12V-to-24V power converter**

### Power Transmitter Unit

The Power Transmitting Unit – PTU – is responsible for a safe delivery of power.

It fulfills all the requirements of the CoaXPress standard for a CoaXPress Host, namely:

- It is capable of delivering up to 17 W of 24 V DC power to the connector
- It implements an over-current protection device – OCP
- It supports the CoaXPress PoCXP detection method

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

- Controlling the PoCXP automatic detection
- Disabling or interrupting the power delivery
- Resetting the OCP when tripped

### 12V Power Input with built-in 12V-to-24 V Power Converter

Applies to: Mono Duo Quad QuadG3 QuadG3DF QuadCXP3 Quad3DLLE  
Octo QuadCXP12

All the PTU's of a PCI Express Coaxlink card are powered by an external 12 V power supply through an on-board 12 V to 24 V power converter on the Coaxlink card.

The external 12 V supply is attached to the **auxiliary power input connector** through a 6-pin PEG cable.

**Note:** *The 12 V 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.

## 24V Power Input

---

Applies to: `Duo104EMB` `Duo104MIL`

All the PTU's of a PCIe/104 Coaxlink card are powered by an external 24 V power supply.

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

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.

## PTU Control Logic

---

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

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)

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`.

## IMPORTANT NOTE



*For users of early versions of the Coaxlink driver*

Since version 3.1 of the Coaxlink Driver, PoCXP powering is enabled at system power-up.

Consequently, when using PoCXP powered camera(s), the application is not anymore required to enable PoCXP powering by issuing a `CxpPoCxpAuto` command.

## 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.*

**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.*

## 3.7. 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.*

## 3.8. 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. For more information, refer to: .



- 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):

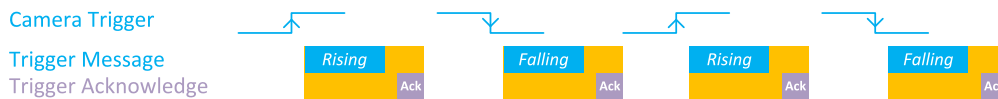
- If the acknowledgement is received before the expiration of the timeout, the transaction terminates normally.
- If no acknowledgement is received within the 20  $\mu$ s timeout, the transmitter performs a **retry**: it resends the trigger packet and initiates a new waiting period for the acknowledgement.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

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

*Default settings*

```
CxpTriggerMessageFormat = Pulse; CxpTriggerAckTimeout = 20.0; CxpTriggerMaxResendCount = 3;
```

*Case 1: CameraControlMethod = RG; camera replies immediately with ACK*



*Case 2: CameraControlMethod = RC; camera replies immediately with ACK*



*Case 3: no reply from camera: retry after timeout (20 us); transaction aborted after 3 retries*



**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 acknowledgement 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 acknowledgement 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 Acknowledgement packet in response to every Trigger packet. If the acknowledgement 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.

#### Alternate settings

*Case 1:* `CameraControlMethod = RG;CxpTriggerMessageFormat = Pulse; CxpTriggerAckTimeout = 0;`



*Case 2:* `CameraControlMethod = RC;CxpTriggerMessageFormat = Pulse; CxpTriggerAckTimeout = 0;`



*Case 3:* `CameraControlMethod = RC;CxpTriggerMessageFormat = Rising; CxpTriggerAckTimeout = 0;`



*Case 4:* `CameraControlMethod = RC;CxpTriggerMessageFormat = Toggle; CxpTriggerAckTimeout = 0;`



### Trigger message transactions using alternate settings to allow higher trigger rates

### Alternate settings for fastest trigger rate

The fastest trigger rate of 297.6 kHz can be achieved when:

- CameraControlMethod = RC (asynchronous reset camera, camera-controlled exposure),
- CxpTriggerAckTimeout = 0 (acknowledge checking disabled),
- CxpTriggerMessageFormat = Rising Or CxpTriggerMessageFormat = Toggle

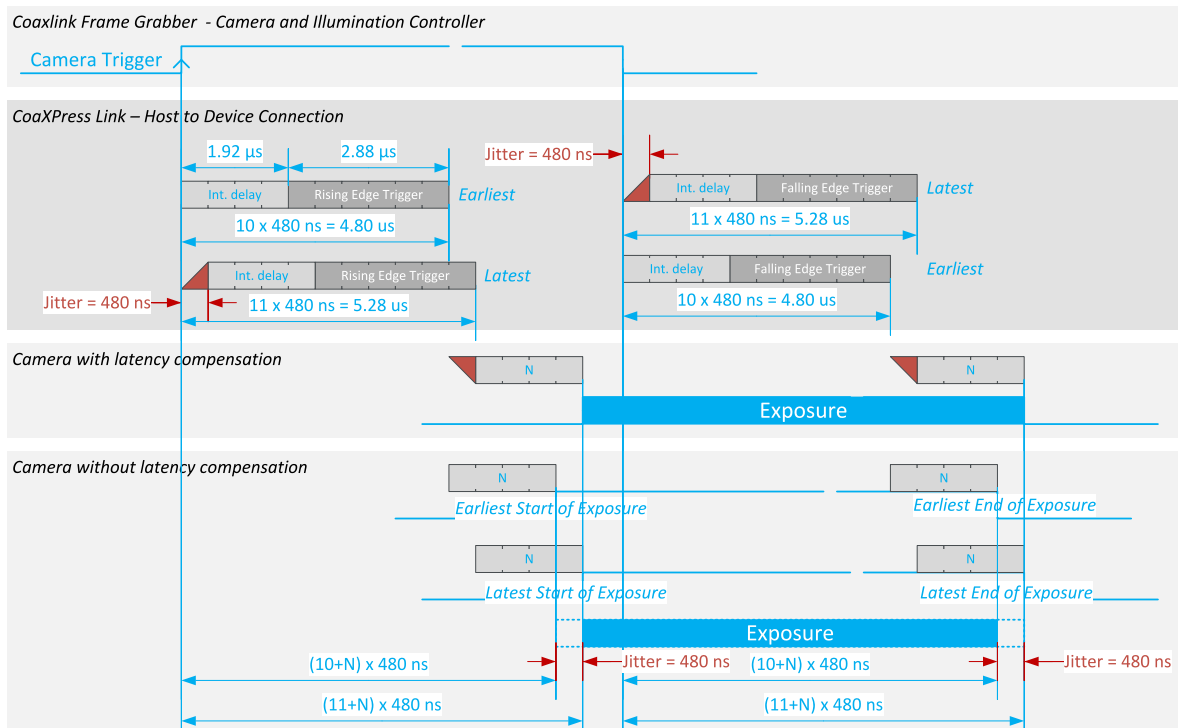
## Camera Trigger to Exposure Latency

### Trigger Accuracy

The Host to Device trigger packets are transmitted over the low-speed (20.833 Mbps) connection of the CoaXPress Link. The transmission of trigger packets can only start at the boundary of a character. This introduces a jitter of 480 nanoseconds corresponding to one character transmission time.

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 2 ns (1/24th of the bit period). 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



CIC Camera Trigger to Sensor Exposure timing diagrams

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  $\mu$ s corresponding to a 4-character pipeline delay in the Trigger Transmitter implementation.
- A fixed delay of 2.88  $\mu$ s 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  $\mu$ s.

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)  $\mu$ s.

## 3.9. CoaXPress Lamps

Each connector of the CoaXPress Host Interface is associated with a *CoaXPress Host Indicator Lamp* that indicates the state of the CoaXPress Link connection.

### CoaXPress Host Connector Indicator Lamps State

Symbol	Lamp State	Meaning
	Off	The Coaxlink card is not powered
	Solid orange	System booting
	Fast flash alternate green / orange	The connection detection is in progress; PoCXP is active. <i>This state is shown for a minimum of 1s even if the connection detection is faster</i>
	Fast flash orange	The connection detection is in progress; PoCXP is off. <i>This state is shown for a minimum of 1s even if the connection detection is faster</i>
	Solid red	The PoCXP over-current protection has tripped.
	Solid green	The Device to Host connection is established, but no data being transferred
	Slow pulse orange	The Device to Host connection is established, but the Host is waiting for a trigger.
	Fast flash green.	The Device to Host connection is established and image data is being transferred

### Flashing Lamp States Timing Definitions

Indication	Timing
Fast flash	<b>12.5Hz @25% duty cycle:</b> 20 ms on, 60 ms off
Fast flash alternate (color 1/color 2)	<b>12.5Hz @25% duty cycle:</b> 20 ms on (color 1), 60 ms off, 20 ms on (color 2), 60 ms off
Slow flash	<b>0.5Hz @50% duty cycle:</b> 1 second on, 1 second off
Slow pulse (red   orange)	<b>1Hz @ 20% duty cycle:</b> 200ms on, 800ms off

## 3.10. 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.*

## 3.11. 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*.

**Note:** For more information about the theory of bit error rate testing, please refer to [http://en.wikipedia.org/wiki/Bit\\_error\\_rate](http://en.wikipedia.org/wiki/Bit_error_rate).

#### Host PC requirements

- The Host PC must be equipped with at least one EURESYS Coaxlink board.
- The Coaxlink driver version 9.3.2 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 since version 9.3.2. 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 than:
  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
1634 Coaxlink Duo PCIe/104-MIL	512 MB
1635 Coaxlink Quad G3 DF	1 GB
1637 Coaxlink Quad 3D-LLE	1 GB
1638 Coaxlink Quad CXP-3	128 MB
3602 Coaxlink Octo	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

Refer to "[Firmware Variants per Product](#)" on page 23 for the available buffer size per data stream and per device.

### 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 `DSSstartAcquisition` and `DSSstopAcquisition` 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 `DSSstartAcquisition` and `DSSstopAcquisition` function calls and, according to the settings of the Image Acquisition Controller, to the Start-of-scan and the End-of-scan triggers.

For more information and configuration instructions, refer to, refer to ["Line Scan Acquisition"](#) on page 206.



## 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.

For more information and configuration instructions, refer to ["Flat Field Correction" on page 155](#) .

### Lookup Table processing

---

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

For more information and configuration instructions, refer to ["Lookup Table Processing" on page 164](#) .

### Bayer CFA decoding

---

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

For more information and configuration instructions, refer to ["Bayer CFA Decoding" on page 178](#) .

### 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.

For more information and configuration instructions, refer to ["Pixel Component Unpacking" on page 62](#).

## Pixel component ordering

---

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

For more information and configuration instructions, refer to ["Pixel Component Re-Ordering"](#) on page 64.

## Endianness conversion

---

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

For more information, refer to ["Endianness Conversion"](#) on page 65.

## 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 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

Input Pixel Format	FFC	Unpacking Mode	RedBlueSwap	Output Pixel Format
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:

- Unpacking to lsb (Default setting)
- Unpacking to msb
- No unpacking

**Note:** For Coaxlink driver versions prior to 4.2, the unpacking operation was not configurable: 10-bit, 12-bit, and 14-bit pixel component data were unpacked to 16-bit using the alignment to msb method.

**Note:** Since Coaxlink driver 4.3, the default option is "Unpacking to lsb".

**Note:** For Coaxlink driver versions prior to 7.0.1, the unpacking operation could not be disabled.

### 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

**Note:** 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*

**Note:** *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.

**Note:** *This option is restricted to monochrome and Bayer CFA pixel formats.*

## 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 `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.

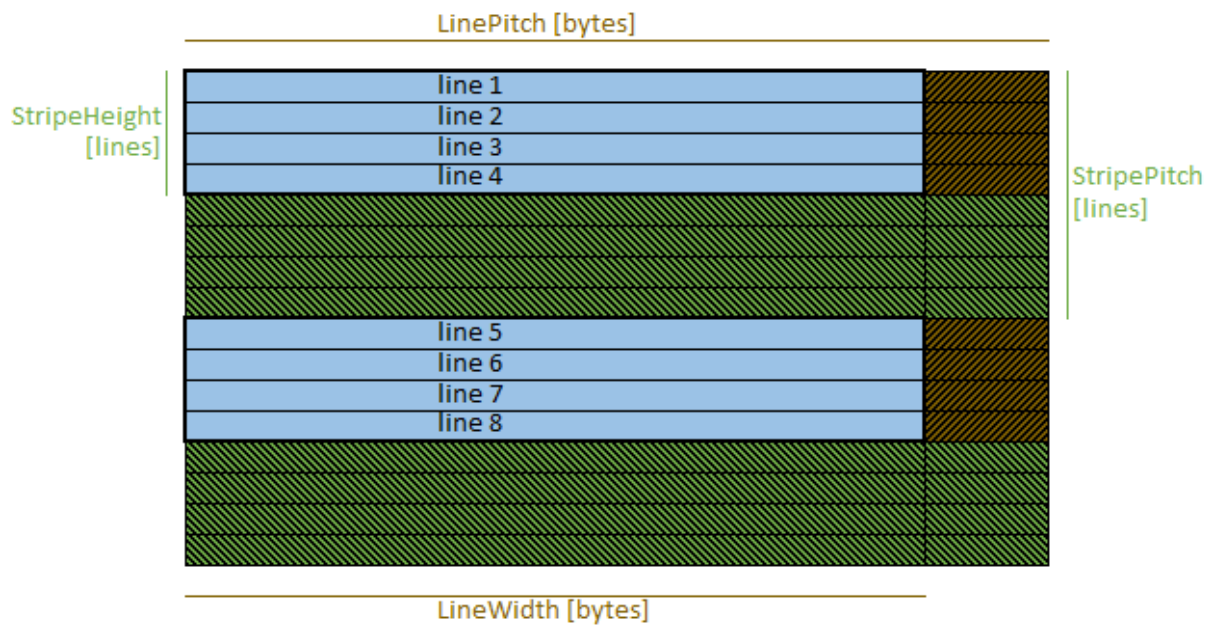


Image buffer padding model

### 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.

The values of `LineWidth` and `LinePitch` can only be changed when the data stream doesn't own any buffer (i.e., before announcing any buffer, or after all announced buffers have been revoked).

`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.

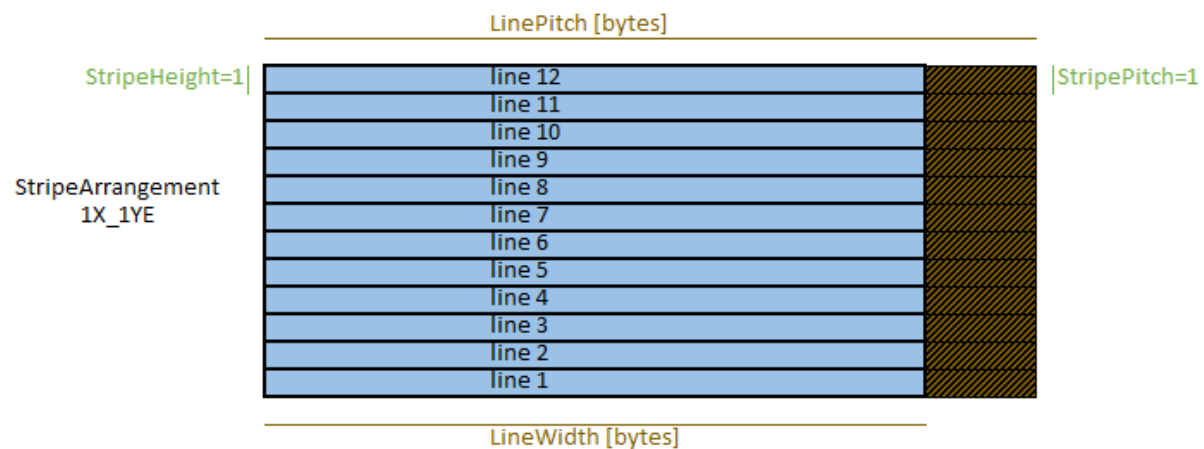
When `StripePitch > StripeHeight`, the stripe padding is enabled: the DMA engine leaves `StripePitch - StripeHeight` lines of padding at the end of each stripe.

The values of `StripeHeight` and `StripePitch` can only be changed when the data stream doesn't own any buffer (i.e., before announcing any buffer, or after all announced buffers have been revoked).

`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:  $\text{BufferBase} + (\text{BufferSize} + \text{LinePitch} - \text{LineWidth}) / \text{LinePitch} * \text{LinePitch} - \text{LinePitch}$ .

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  $\text{BufferBase} + n * \text{LinePitch}$ ;
- 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 zero, `LineWidth` will be used instead. Similarly, if `StripeHeight` is zero, 1 will be used instead.

## 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 sampling method 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 `DSSstartAcquisition()` function call until now, if the acquisition is still active otherwise until the last `DSSstopAcquisition()` 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 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:

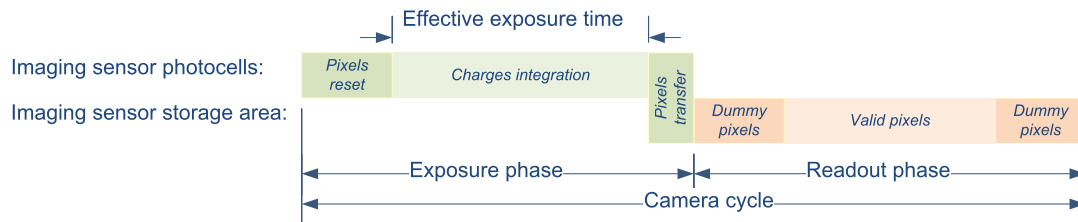
- The number of filled GenTL buffers and the corresponding number of frames (area-scan) or lines (line-scan)
- The number of transferred bytes of image data.

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

# 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).



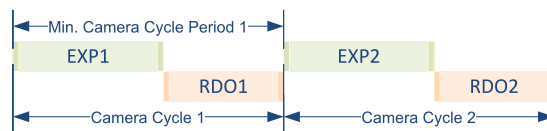
Some sensors provide the capability to select one or more region of interest (ROI) speeding up the readout since less data needs to be transferred.

## 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.



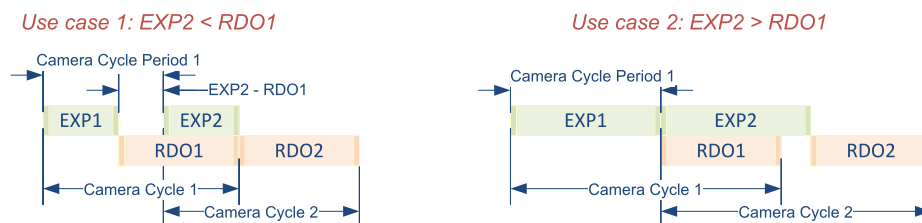
#### Shortest possible cycle period achievable by cameras NOT allowing the cycle overlapping

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

**Note:** Only a minority of industrial cameras are NOT allowing the cycle overlapping!

### Rules for cameras allowing overlapping

- The exposure phases of two consecutive camera cycles may NEVER overlap.
- 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:

$$\text{Min Cycle Period}_n = \text{EXP}_n$$

**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

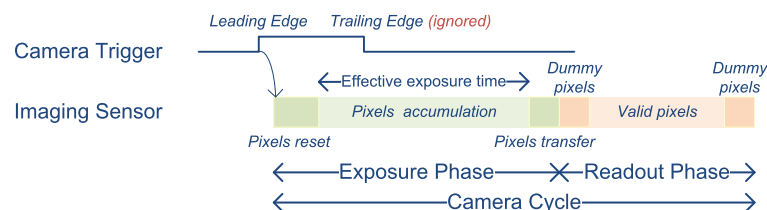
- Free-run cameras not using any external trigger signal
- Asynchronous-reset cameras using an external trigger signal not delivered by the frame grabber.

**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.



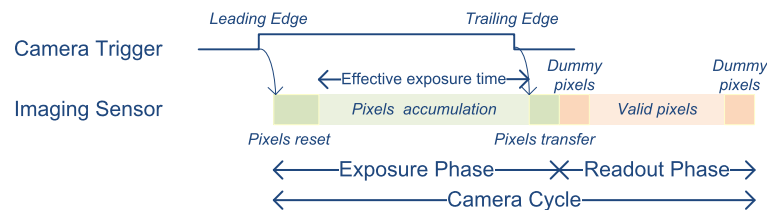
#### Grabber controlled camera cycle using the RC camera control method

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.*

**Important:** *The external controller is entirely responsible for the camera cycle timings!*

# 6. Illumination Control Principles

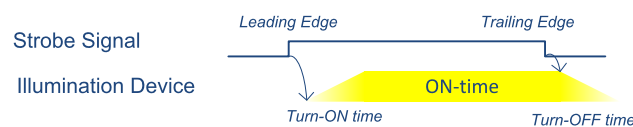
## 6.1. Illumination Devices

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.



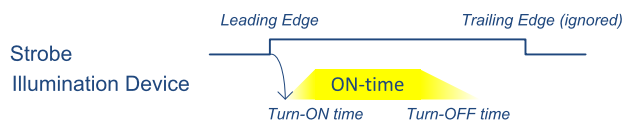
#### Timing diagram for intermittent illumination devices

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

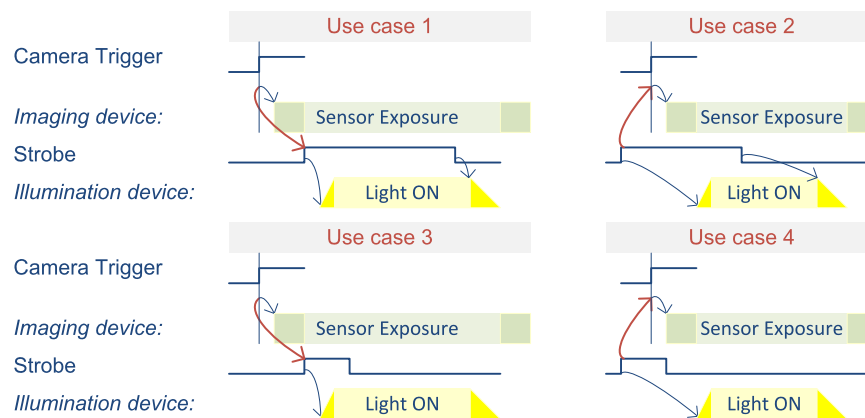
The on-time duration is either uncontrolled or controlled by the illumination device itself.

**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.

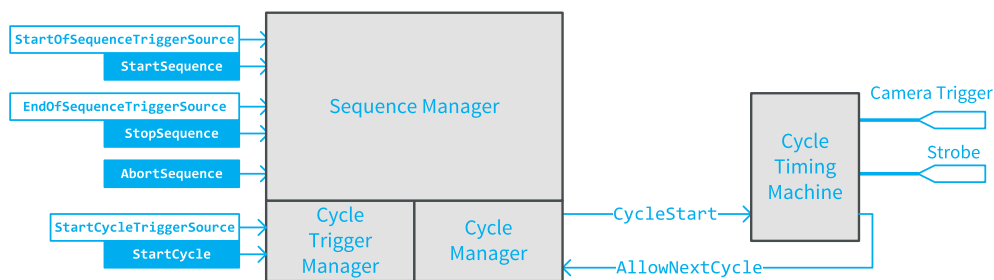
# 7. Camera and Illumination Controller

## 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.

For more information refer to "[CIC Output Signals Routing](#)" on page 89.

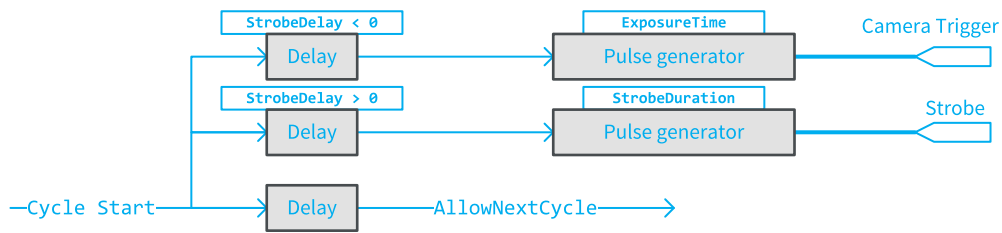


**Camera and Illumination Controller block diagram**

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**. For more information refer to "[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. For more information refer to "[Cycle Manager](#)" on page 85
- 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**. For more information refer to "[Cycle Trigger Manager](#)" on page 86.
- The **Sequence Manager** manages sequences of CIC cycles according to user-defined start sequence and stop sequence conditions. For more information refer to "[Sequence Manager](#)" on page 88.

## 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.
- `StrobeDuration` 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`.

Refer to "[Cycle Timing Diagrams](#)" on page 90 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!*

Refer to "[Consecutive Overlapping CIC Cycles](#)" on page 92 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 `CycleTriggerSource≠Immediate` AND `CycleMaxPendingTriggerCount=0`

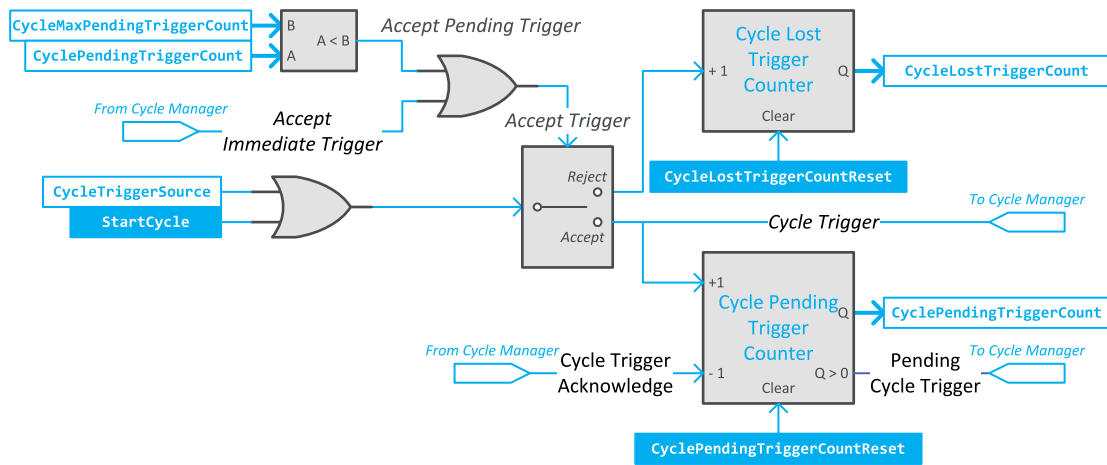
### Pending Trigger Condition

---

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

This condition applies only when `CycleTriggerSource≠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.

It 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.

## 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

For detailed information about the Host to Device Trigger transmitter, refer to "[CoaXPress Host To Device Trigger](#)" on page 40.

#### 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.

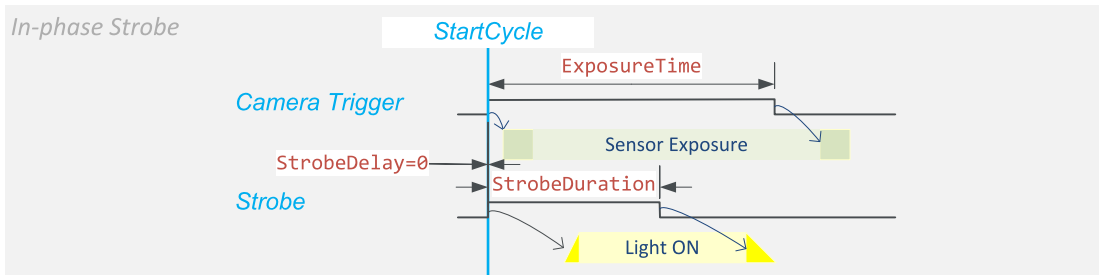
## 7.7. CIC Timing Diagrams

Cycle Timing Diagrams .....	90
Consecutive Overlapping CIC Cycles .....	92
Cycle Sequence Timing Diagrams .....	97

# 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`.

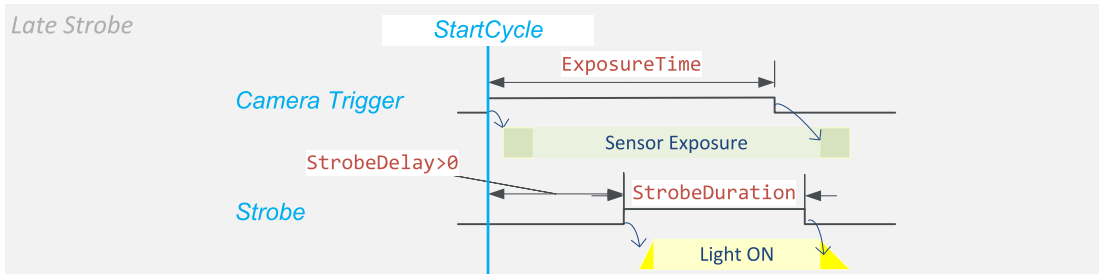
## In-phase Strobe Case



**StrobeDelay = 0**

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

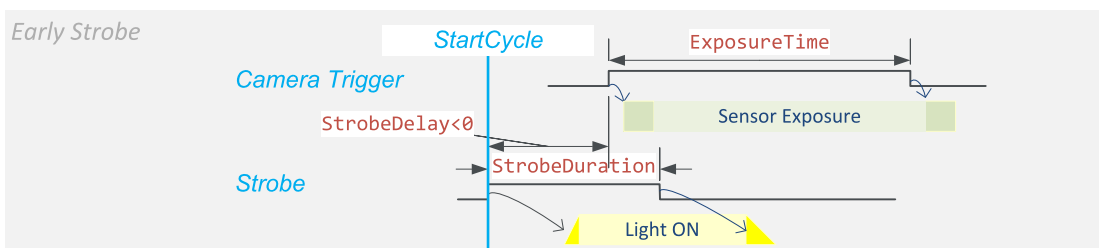
## Late Strobe Case



**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  $- \text{StrobeDelay}$  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.

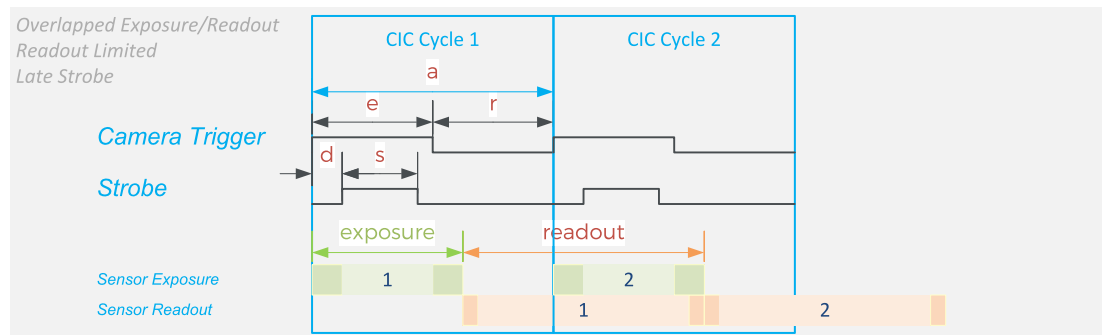
The "readout-limited" use cases illustrate situations where the cycle period is equal to the duration of the readout phase.

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.

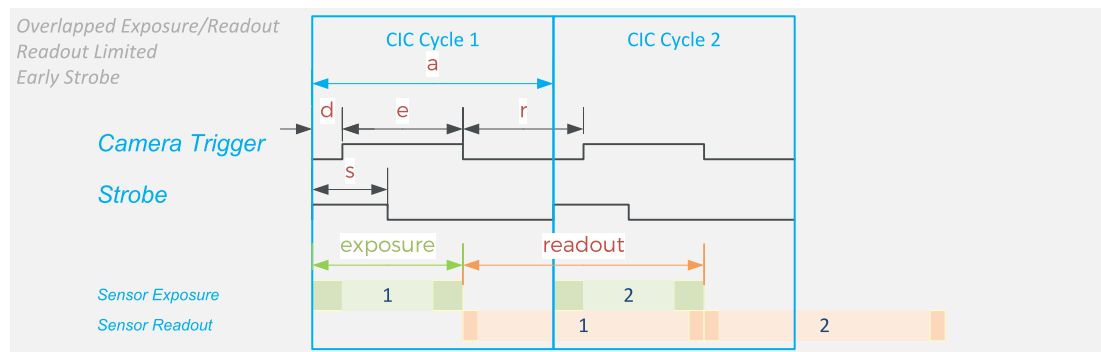
In that situation:

- $r$  is likely larger than `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  $e$  (= Exposure Time) is significantly smaller than the readout duration.

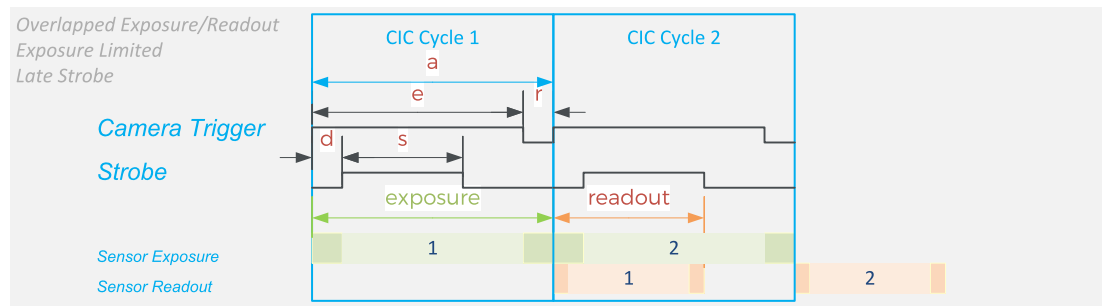
In that situation:

- $r$  is likely larger than `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  $(d + e < \text{readout duration})$ .
- 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

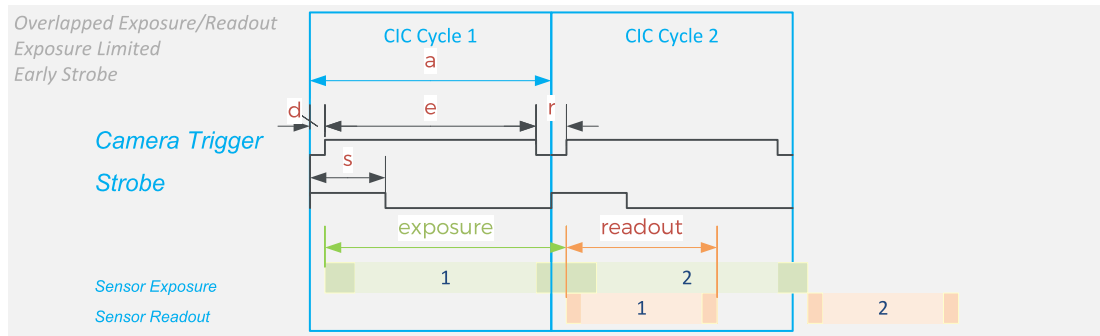
This situation occurs when  $e$  (= Exposure Time) is significantly larger than the readout duration.

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 in case of small negative `StrobeDelay` values.

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 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.

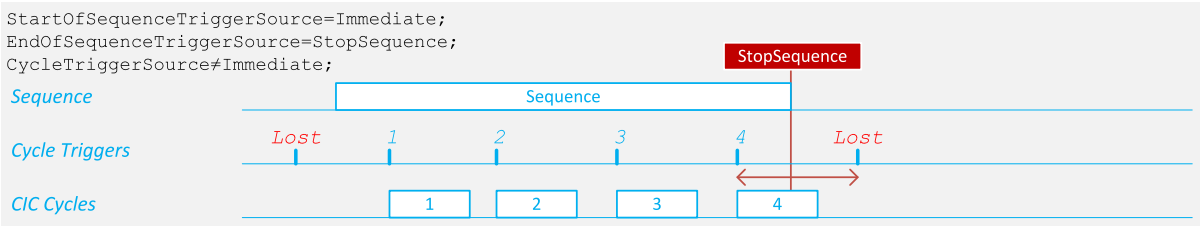
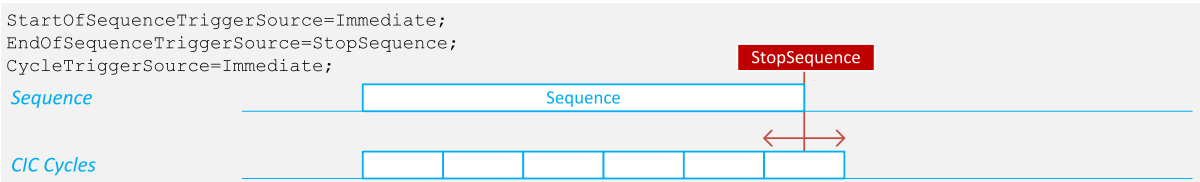
The user must set `ExposeRecovery` to a value corresponding to the largest of the following two values:

- Minimal time interval allowed by the camera between consecutive `Camera Trigger` pulses.
- - `StrobeDelay`

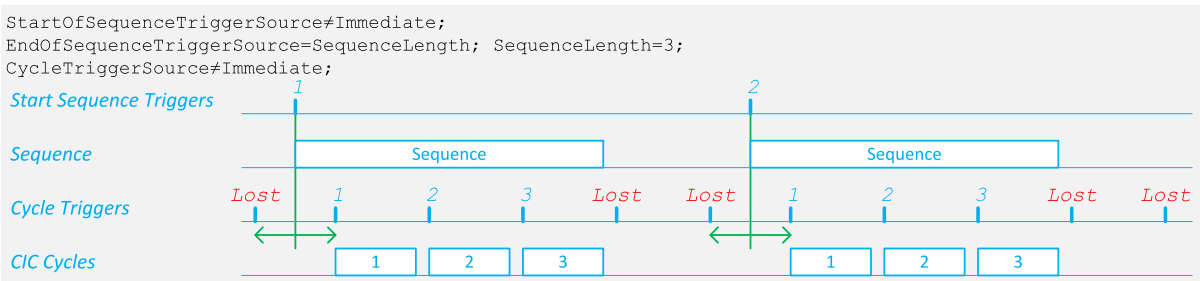
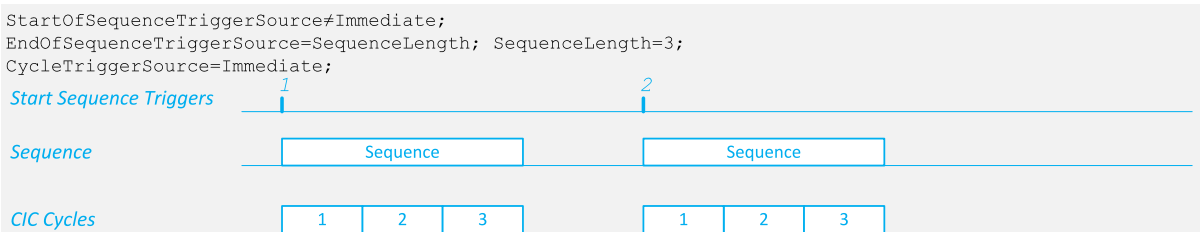
**Note:** When `CycleTriggerSource = Immediate`, the cycle rate can be lowered to the desired rate by assigning a greater value to `TargetFramePeriod`.



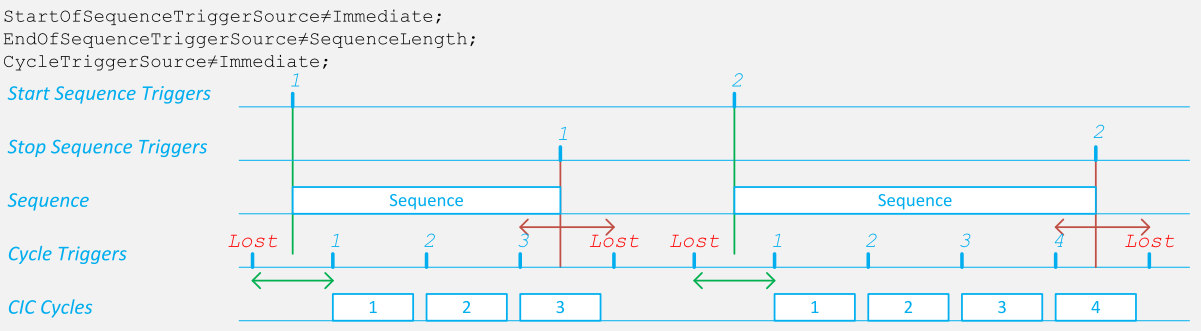
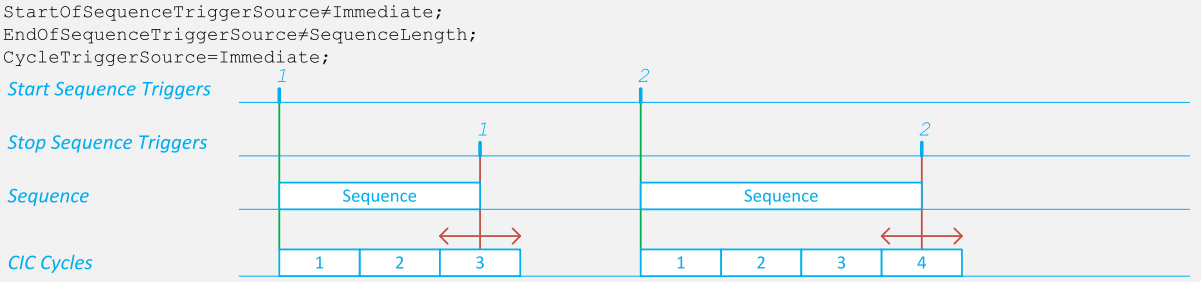
# Cycle Sequence Timing Diagrams



## 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 Line Types

I/O Line Type	Electrical Style	Frequency range	Name Prefix
Differential input	RS-422 differential inputs	Up to 5 MHz	DIN
TTL input/output	5 Volt-tolerant LVTTTL single-ended input or LVTTTL (totem-pole, open-emitter and open-collector) single-ended output	Up to 5 MHz	TTLIO
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/30V contact output	Up to 50 kHz	IOUT

## 8.2. Available I/O Lines

### I/O Lines per Product

Product	I/O Set 1	I/O Set 2
1629 Coaxlink Duo PCIe/104-EMB	OK*	-
1630 Coaxlink Mono	OK	-
1631 Coaxlink Duo	OK	OK
1632 Coaxlink Quad	OK	OK
1633 Coaxlink Quad G3	OK	OK
1634 Coaxlink Duo PCIe/104-MIL	OK*	-
1635 Coaxlink Quad G3 DF	OK	OK
1637 Coaxlink Quad 3D-LLE	OK	OK
1638 Coaxlink Quad CXP-3	OK	OK
3602 Coaxlink Octo	OK	-
3603 Coaxlink Quad CXP-12	OK	

**Important:** (\*) For **Duo104EMB** and **Duo104MIL** : the I/O Line Set 1 is available only when the 3300 HD26F I/O module for Coaxlink Duo PCIe/104, accessory is installed.

### I/O Set 1

The I/O Line Set 1 is composed of 10 I/O:

I/O Type	I/O Line Names	Count
Differential input	DIN11, DIN12	2
TTL inputs/output	TTLIO11, TTLIO12	2
Isolated input	IIN11, IIN12, IIN13, IIN14	4
Isolated output	IOUT11, IOUT12	2

### I/O Set 2

The I/O Line Set 2 is composed of 10 I/O lines.

I/O Line Type	I/O Line Names	Count
Differential input	DIN21, DIN22	2
TTL inputs/output	TTLIO21, TTLIO22	2
Isolated input	IIN21, IIN22, IIN23, IIN24	4
Isolated output	IOUT21, IOUT22	2

## 8.3. I/O Lines Usage

### Input Lines

---

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

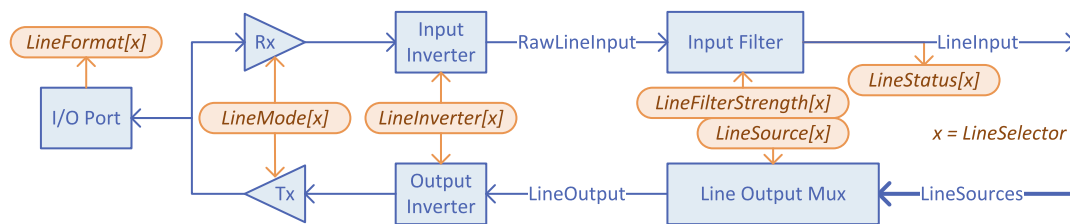
- General purpose output: An output signal whose state can be set by the host application.
- Strobe output: An output signal usually used to control a strobe light, in synchronization with the camera.
- Camera trigger output (only available on TTLIO): An output signal generated by the Coaxlink card and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

# 8.4. I/O Control Blocks

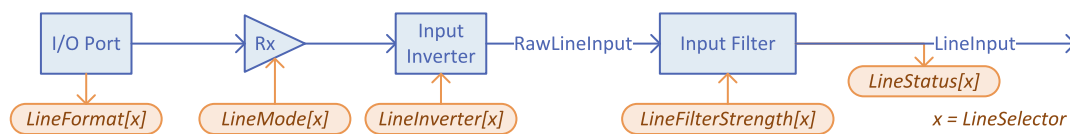
Every I/O line is controlled through one I/O control block.

## Block Diagrams

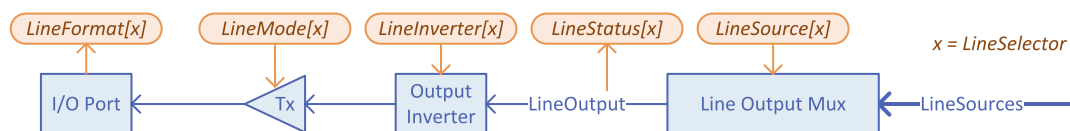
There are 3 variations of the control block depending on the input/output capabilities of the I/O line:



**I/O control block for bidirectional I/O lines**



**I/O control block for input-only I/O lines**



**I/O control block for output-only I/O lines**

In the above drawings:

- 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

## 8.5. Line Mode Control

For all the I/O lines, the `LineMode` feature reports the configuration of the I/O port.

Only the TTL I/O lines have a user-configurable line mode.

A TTLIO can operate as an input or as an output.

When used as input, no configuration is required since at power-up all TTLIO ports are configured as Inputs.

When used as output, the user has the choice between 3 configurations of the line driver. The line driver can be configured in 3 ways::

- As a totem-pole driver capable of driving low and high.
- As an open-collector driver capable of driving low only
- As an open-emitter driver capable of driving high only.

The two latest configurations allows wired-AND configurations.

## 8.6. I/O Lines Specifics

Line Selector	Bit#	Line Format	Line Mode
DIN11	0	DIFF	Input
DIN12	1	DIFF	Input
DIN21	2	DIFF	Input
DIN22	3	DIFF	Input
IIN11	4	ISO	Input
IIN12	5	ISO	Input
IIN13	6	ISO	Input
IIN14	7	ISO	Input
IIN21	8	ISO	Input
IIN22	9	ISO	Input
IIN23	10	ISO	Input
IIN24	11	ISO	Input
IOUT11	12	ISO	Output
IOUT12	13	ISO	Output
IOUT21	14	ISO	Output
IOUT22	15	ISO	Output
TTLIO11	16	TTL	Input, Output, DriveLow or DriveHigh
TTLIO12	17	TTL	Input, Output, DriveLow or DriveHigh
TTLIO21	18	TTL	Input, Output, DriveLow or DriveHigh
TTLIO22	19	TTL	Input, Output, DriveLow or DriveHigh

The above table summarizes 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`

**Note:** Check the "Available I/O Lines" on page 101 for applicable values.

## 8.7. 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.*

## 8.8. 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 $\mu$ s
Medium	200 ns	200 ns	2 $\mu$ s
High	500 ns	500 ns	5 $\mu$ s
Highest	1 $\mu$ s	1 $\mu$ s	10 $\mu$ s

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.

## 8.9. 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

## 8.10. 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.

## 8.11. 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. <i>Line may be left unconnected or connected with the wrong polarity.</i>
		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.



## 8.12. 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	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.

## 8.13. 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

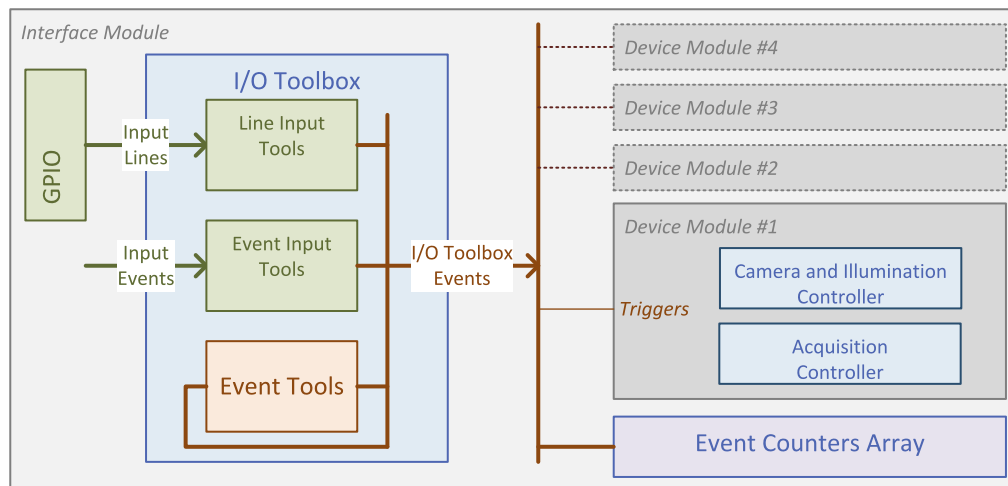
The opto-coupler is OFF.

# 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.

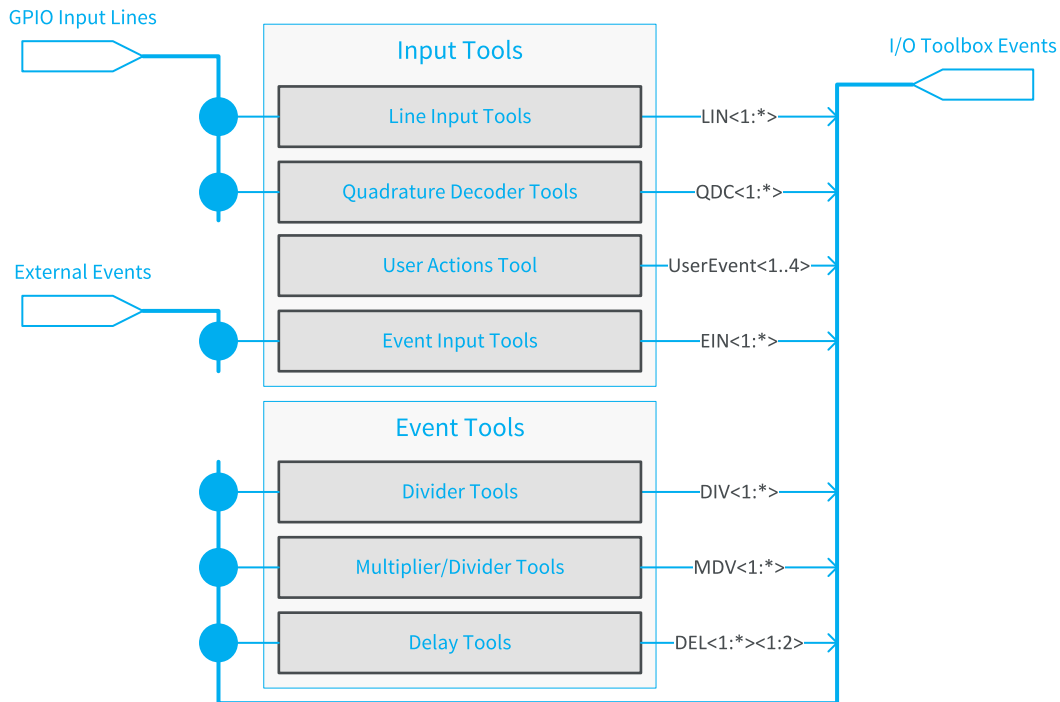
Any I/O Toolbox events can be used in any GenTL Device module of the card as:

- 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	-

### 1631 Coaxlink Duo

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

### 1632 Coaxlink Quad

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

### 1633 Coaxlink Quad G3

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

### 1629 Coaxlink Duo PCIe/104-EMB1634 Coaxlink Duo PCIe/104-MIL

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

### 1635 Coaxlink Quad G3 DF

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

### 1638 Coaxlink Quad CXP-3

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

### 3602 Coaxlink Octo

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

### 3603 Coaxlink Quad CXP-12

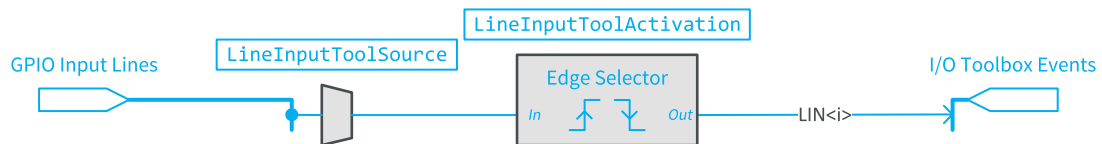
Firmware Variant	#LIN	#QDC	#DIV	#MDV	#DEL	#UAS	#EIN
1-camera	8	1	1	1	1	1	-



## 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: <b>LIN&lt;i&gt;</b>

### 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

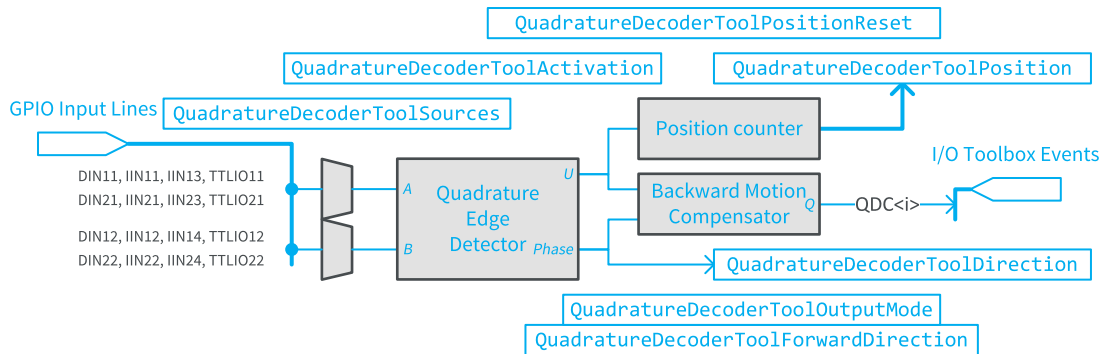
The Line Input tool detects the rising or the falling edge of the *LineInput* signal delivered by the I/O Control block selected by `LineInputToolSource`.

The Line Input tool delivers one event at every rising or falling edge 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: <b>QDC&lt;i&gt;</b> 1 status bit: <b>QDC&lt;i&gt;Direction</b> 1 status word: <b>QDC&lt;i&gt;Position</b>

### 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.

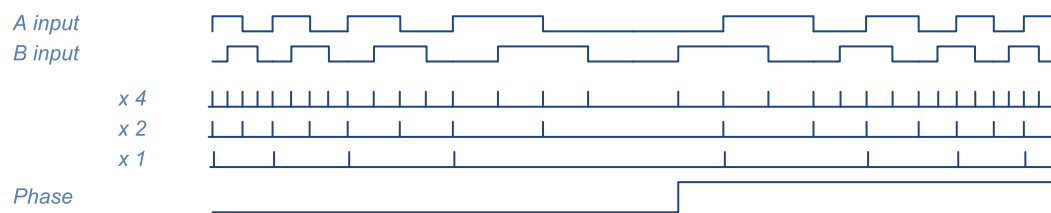
## Operation

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**

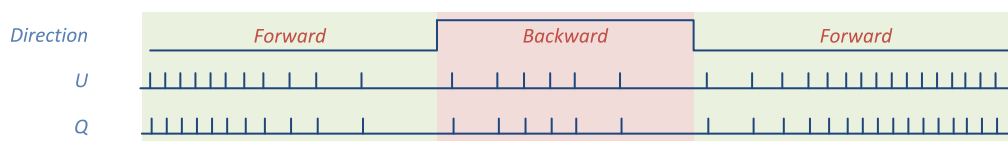
The U stream may be filtered by the backward motion compensator before being delivered to the QDC<i> output.

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

### Backward motion compensator

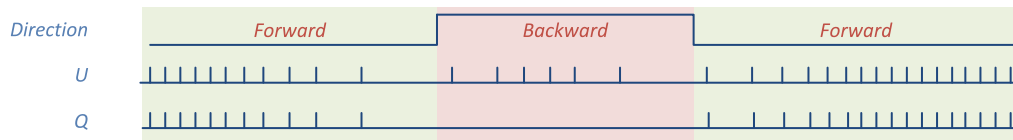
The **backward motion compensator** (BMC) filters the U stream according to the `QuadratureDecoderToolOutputMode` setting.

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



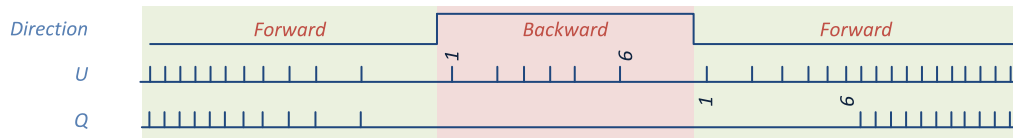
**BMC waveforms - Unfiltered**

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.

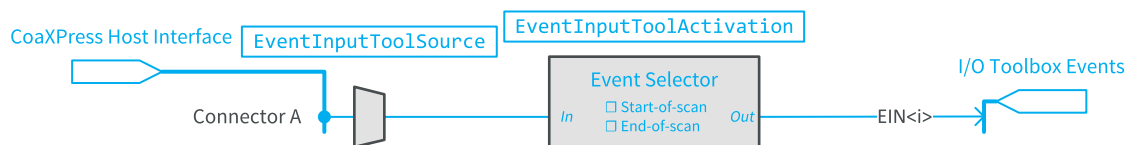
The counter can be reset using the `QuadratureDecoderToolPositionReset` command.

## 9.5. 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: <b>EIN&lt;i&gt;</b>

### 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

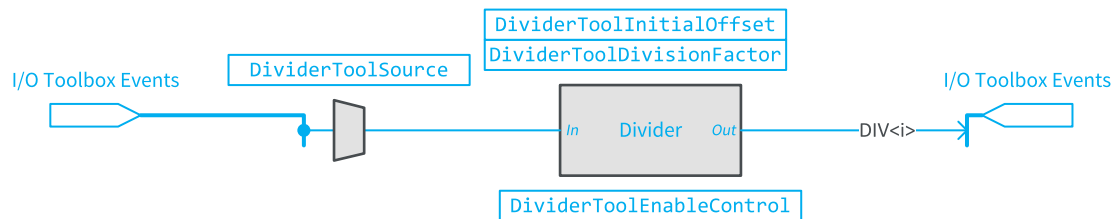
The Event Input tool decodes the custom CoaXPress GPIO message received on the CoaXPress connector A of a slave 1635 Coaxlink Quad G3 DF.

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.

## 9.6. Divider Tool

Tool Name	Short Name	Inputs Count/Type	Outputs Count/Type/Name
Divider Tool	DIV	1 Toolbox event stream	1 event stream: <b>DIV&lt;i&gt;</b>

### Diagram

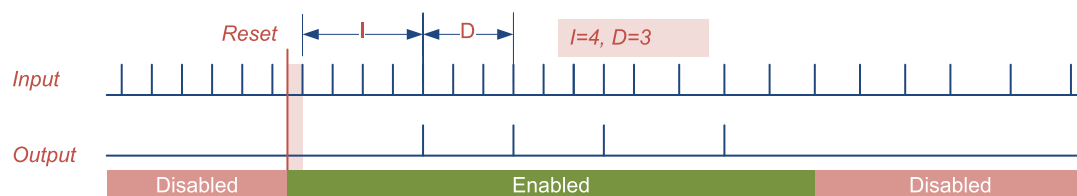


**DIV tool functional and wiring diagram**

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.

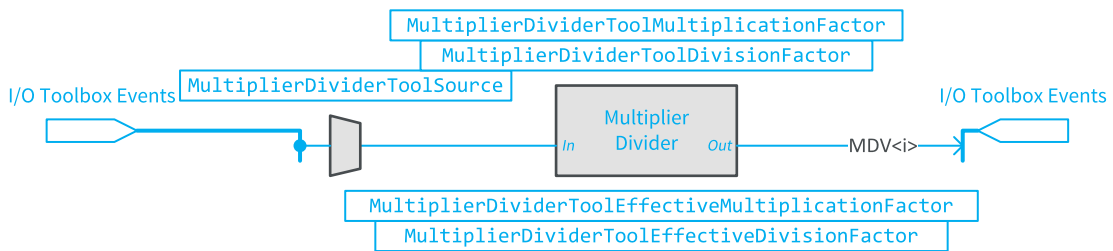
The initial offset – **I** – is defined by `DividerToolInitialOffset`. The default value is 0 and the value range is 0 ... 65535.

The operation state is defined by `DividerToolEnableControl`. The default value is `Disable0`

# 9.7. 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: <b>MDV&lt;i&gt;</b>

## Multiplier/Divider Tool Wiring Diagram



**MDV tool functional and wiring diagram**

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

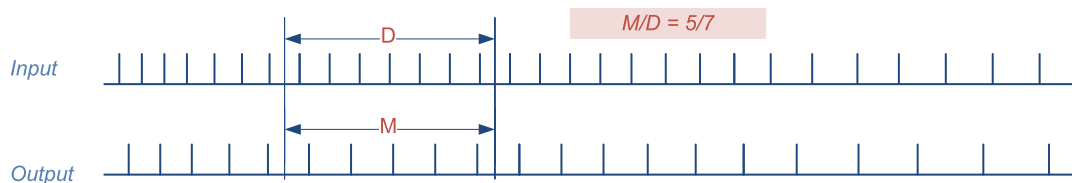
The tool feeds one I/O Toolbox event stream named MDV<i>.

## 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.

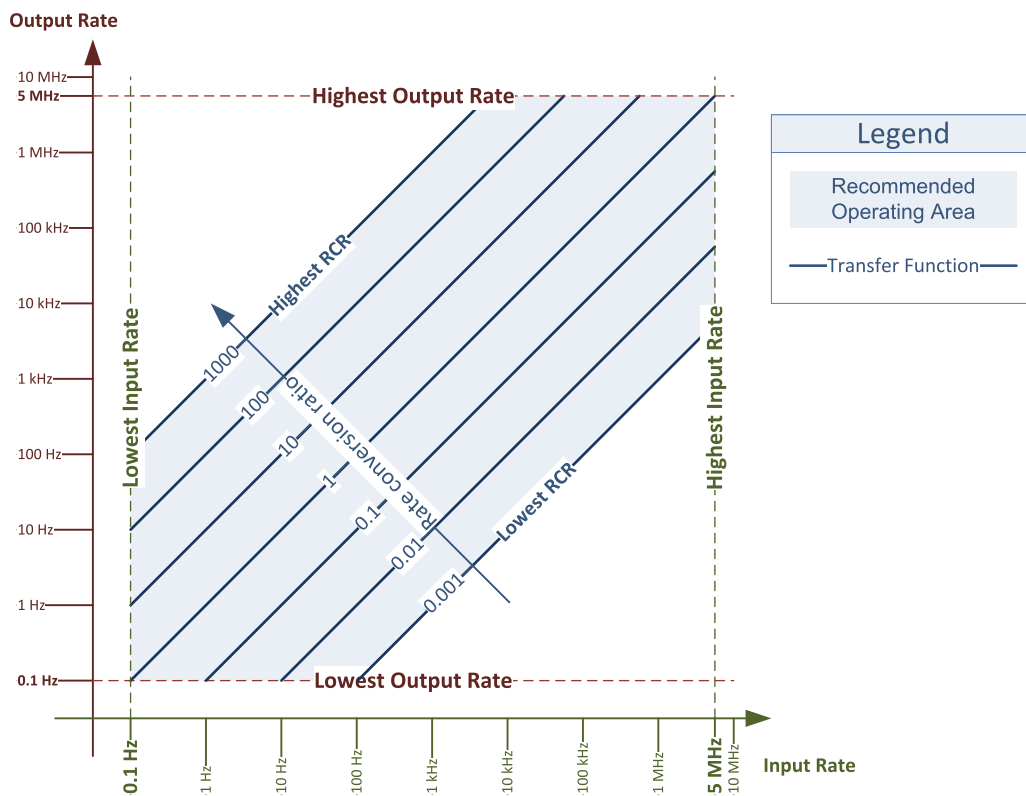
The effective multiplication and division factors are respectively reported by `MultiplierDividerToolMultiplicationFactor` and `MultiplierDividerToolDivisionFactor`.

**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



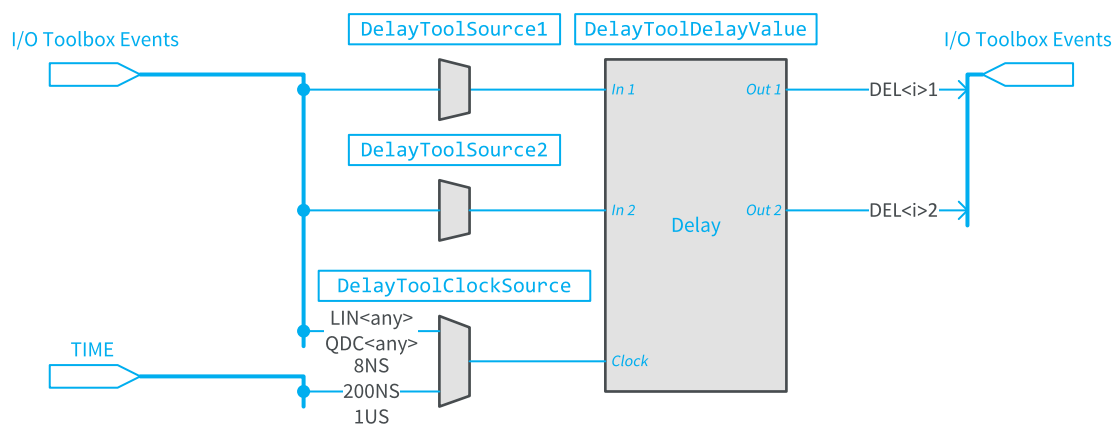


## MDV tool operating limits diagram

## 9.8. 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: <b>DEL&lt;i&gt;1</b> , <b>DEL&lt;i&gt;2</b>

### 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.

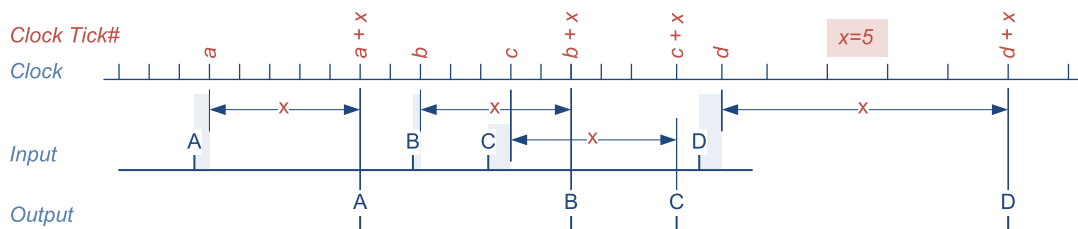
The same clock source applies to both channels. The clock source is defined by DelayToolClockSource. It can be a **time base** or a **line tool event stream**.

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**.

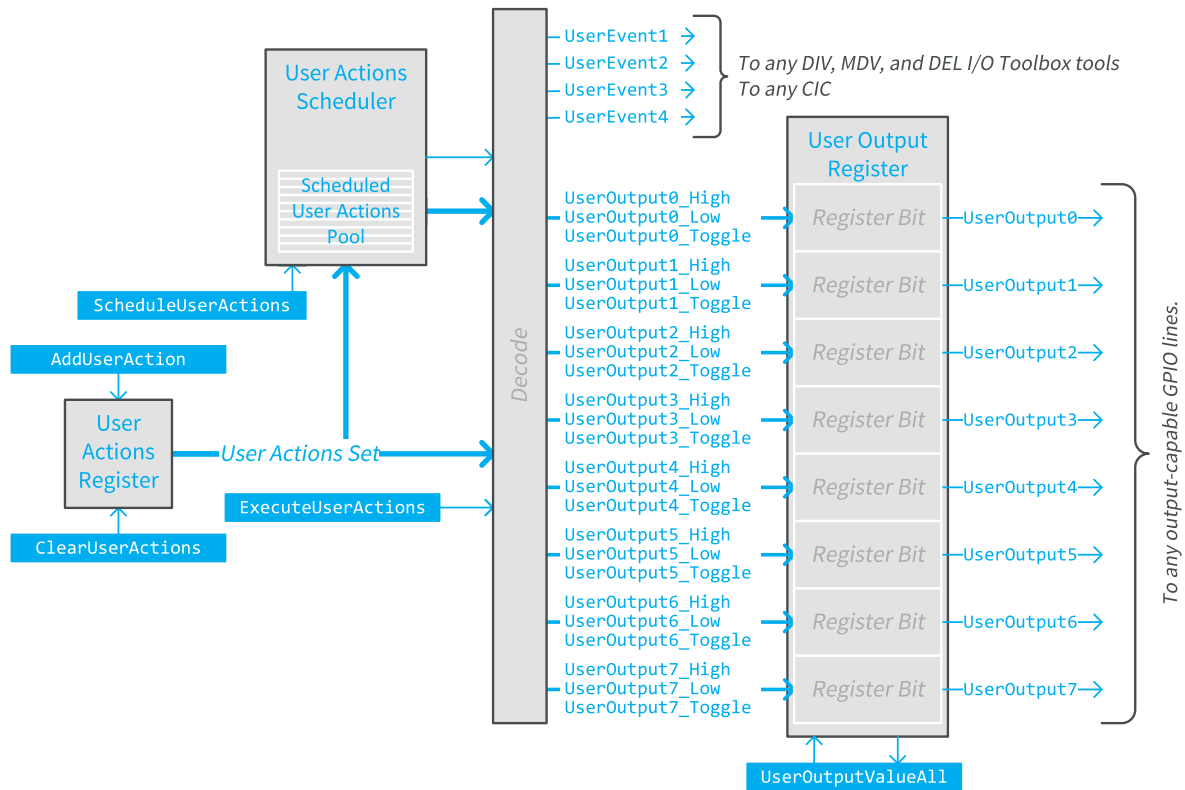
**Important:** *The Delay tool operates as a delay line. The tool may accept a new event while the previous one is not yet delivered! The Delay tool is capable of recording, globally for all channels, up to 16 distinct events.*



**DEL tool waveforms**

## 9.9. 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**:

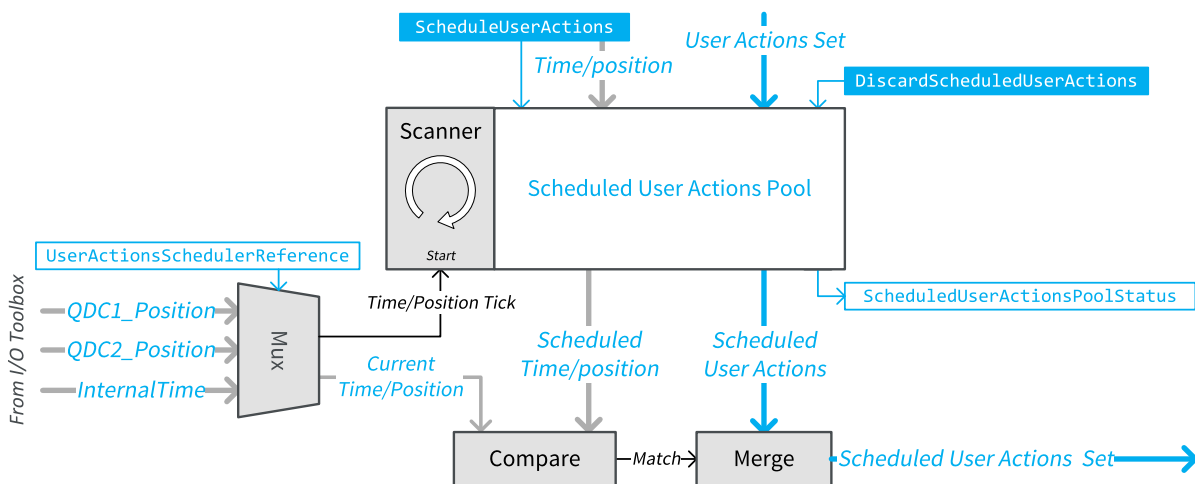
- Generate events on any User Events source.
- Setting high any bit of the User Output Register.
- Setting low any bit of the User Output Register
- Toggling any bit of the User Output Register.

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.

**Important:** 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`.

## Schedule 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 as a 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.

# 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.

For an exhaustive list, refer to ["Custom Events Sources" on page 139](#)

### 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 all 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. Refer to ["Event Specific Context Data" on page 142](#)

## 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

Event Source	Description
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 119 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_	A CRC error is detected on the Connection C of the CoaXPress

Event Source	Description
ERROR_CXP_C	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

Bit#	Description
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. For an extensive description, refer to section 4.2 starting on page 34 of the [GenICam GenTL Standard Version 1.4](#).

### 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 `GCRRegisterEvent()` 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.

**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

The **Sub-link Acquisition** feature of 1633 Coaxlink Quad G3 allows to acquire images from specific **8-connection CoaXPress cameras**.

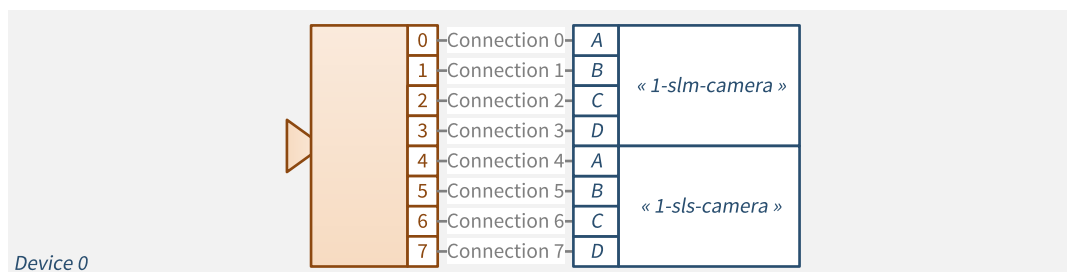
### Camera requirements

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).

### Principles

The 8-connection CoaXPress link is divided into 2 sub-links. Each sub-link connects to a 1633 Coaxlink Quad G3 using the 1D8SL4 connection scheme:



#### 8-connection camera using 2 sub-links and 2 1633 Coaxlink Quad G3

The first 4-connection sub-link connects to the "sub-link-master grabber": a 1633 Coaxlink Quad G3 fitted with the `1-slm-camera` firmware variant.

The next 4-connection sub-link connects to the "sub-link-slave grabber": a 1633 Coaxlink Quad G3 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.

## 11.2. Multi-Stream Acquisition

Applies to: QuadG3

### 4-data-stream Concurrent Acquisition

---

The **1-camera, 4-data-stream** firmware variant of 1633 Coaxlink Quad G3 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**

# 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.

Refer to "[Line-scan Triggers Synchronization](#)" on page 153 for more info.

## 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.

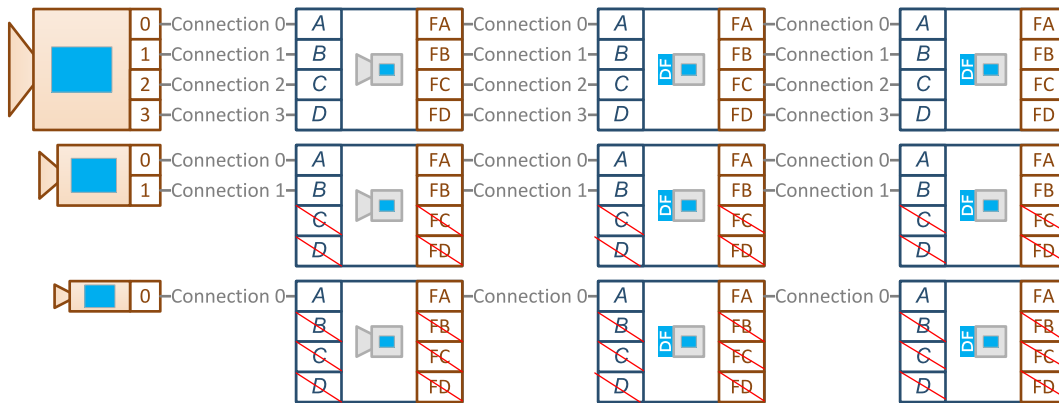
Refer to "[CoaXPress Host To Device Trigger](#)" on page 40 for more details.

# 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.

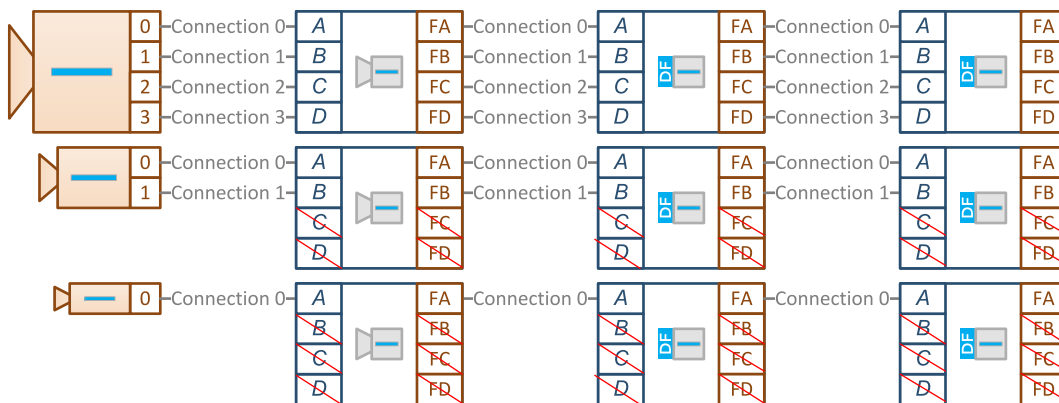
The first 1635 Coaxlink Quad G3 DF card must be configured with a `1-camera` firmware variant; the other must be configured with the `1-df-camera` firmware variant.



## 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. Refer to "[Line Scan Acquisition](#)" on [page 206](#) for more details.

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 125](#) 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 Octo

# What is Flat Field Correction?

The Flat-field correction (FFC) is a method used to correct:

- the differences of light sensitivity between the pixel sensors of a camera
- the differences of illumination intensities in the field-of-view
- the differences in the transmission of light through the lens (for instance: 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:

$$\text{CorrectedPixel} = (\text{RawPixel} - \text{Offset}) * \text{Gain}$$

where both `Offset` and `Gain` coefficients are specific values for each pixel.

## Calibration Procedure

---

The calibration procedure evaluates the `Offset` and `Gain` FFC coefficients of each pixel by computing pixel data of two reference images: the *Dark Image* and the *Flat Image*

**Important:** *the calibration procedure must be redone if any part of the system is changed, including the camera unit, lighting or optics equipment.*

### Dark Image Acquisition

The dark image (a.k.a. dark frame) is an image captured in the dark. Such 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. To obtain the dark image:

1. Cover the lens with a cap.
2. Acquire several images and compute the average response for each pixel of the array

### Flat Image Acquisition

The flat image is an image of a uniform target covering the whole field-of-view. To obtain the flat image:

1. Place a uniform target (white for color cameras) wide enough to cover the whole field-of-view
2. Adjust the lens aperture and the illumination intensity to obtain the brightest possible image providing that no pixels are saturated.
3. Acquire several images and compute the average response for each pixel of the array

## FFC Offset Coefficients Calculation

For each pixel, the `Offset` coefficient is actually the pixel value of the *Dark Image*.

**Note:** *This correction is also called dark-frame subtraction.*

## FFC Gain Coefficients Calculation

The *Dark Image* and the *Flat Image* provide enough data to compute the `Gain` coefficients 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(Dark)`). 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.

## Calibration Procedure 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 **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

## Feature Availability

---

The following table lists the product/firmware variant combinations supporting FFC and their main characteristic:

Product / Firmware variant	Firmware Variant	CoaXPress Interface	DRAM Partitions
1633 Coaxlink Quad G3	1-camera	CXP-6 DIN4	512 MB for FFC coefficients storage 512 MB for FIFO data buffer
3602 Coaxlink Octo	2-camera	CXP-6 DIN4	512 MB for FFC coefficients storage 512 MB for FIFO data buffer

## 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	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:

1. guide the user through the calibration procedure;
2. provide a technical and practical translation of what's described in this document;
3. 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 following help message is displayed when the flag `--help` is given.

```
> ffc-wizard.exe --help
Flat Field Correction Wizard
ffc-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)
  --roi_width=INT    Width of ROI (default: whole image)
  --roi_height=INT   Height of ROI (default: whole image)
  --balance          Compute flat image average on all components rather than on each
component
  --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
<code>--if, --def, --ds</code>	the indexes of the GenTL modules that identify the data stream to use (and/or to configure)
<code>--average</code>	the number of images to acquire for dark and flat acquisitions; only the average of those acquisitions is further used in the calibration procedure
<code>--roi_x, --roi_y, --roi_width, --roi_height</code>	an optional rectangular region to consider when computing the average pixel value of the flat image ( <code>average(Flat)</code> ); this impacts the evaluation of the gain value for each pixel
<code>--balance</code>	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)
<code>--dark-setup, --flat-setup</code>	optional configuration scripts to run before dark and flat acquisitions
<code>--dark-histogram, --flat-histogram</code>	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
<code>--no-interact</code>	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)

## 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	<code>acquireImages</code>
acquiring a series of flat images to build an average flat image	<code>acquireImages</code>

Task	Done by function
computing the <code>gain</code> values for each pixel component	<code>computeGain</code>
using the dark pixel component values as <code>offset</code> values	<code>computeOffset</code>
packing <code>offset</code> and <code>gain</code> values into 16-bit little-endian values	<code>packCoefficients</code>
writing the packed coefficients into a binary file	<code>savePackedCoefficients</code>

## 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

### Introduction to LUT Processing

The Coaxlink driver provides lookup table capabilities on a selected set of product/firmware combinations. For an exhaustive list, refer to "[LUT Processing Availability](#)" on the next page.

This release of the Coaxlink driver provides lookup table processing for monochrome pixel formats exclusively. Refer to "[Monochrome Lookup Table Processing](#)" on page 167 for a detailed description.

The Coaxlink driver provides four methods to define the content of lookup tables. For more information, refer to "[LUT Content Definition](#)" on page 168.

To setup lookup tables, refer to "[LUT Setup Procedure](#)" on page 175

## LUT Processing Availability

This topic lists the supported LUT configurations for all firmware variants and products.

### 1630 Coaxlink Mono

Firmware Variant	LUT Configurations
1-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16

### 1631 Coaxlink Duo

Firmware Variant	LUT Configurations
1-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
2-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
1-camera, line-scan	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
2-camera, line-scan	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16

### 1632 Coaxlink Quad

Firmware Variant	LUT Configurations
1-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
2-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
1-camera, line-scan	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16

### 1633 Coaxlink Quad G3

Firmware Variant	LUT Configurations
1-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
2-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
4-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
1-camera, line-scan	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
2-camera, line-scan	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
4-camera, line-scan	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
1-camera, 4-data-stream	<i>No LUT processing</i>
1-slm-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
1-sls-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16

### 1629 Coaxlink Duo PCIe/104-EMB and 1634 Coaxlink Duo PCIe/104-MIL

Firmware Variant	LUT Configurations
1-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
2-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
1-camera, line-scan	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16

### 1635 Coaxlink Quad G3 DF

Firmware Variant	LUT Configurations
1-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
1-df-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
1-camera, line-scan	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
1-df-camera, line-scan	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16

### 1638 Coaxlink Quad CXP-3

Firmware Variant	LUT Configurations
4-camera	<i>No LUT processing</i>

### 3602 Coaxlink Octo

Firmware Variant	LUT Configurations
1-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
2-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
4-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
5-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
8-camera	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
1-camera, line-scan	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16
2-camera, line-scan	M_8x8, M_10x8, M_10x10, M_10x16, M_12x8, M_12x12, M_12x16

# 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  $\gamma$  - 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

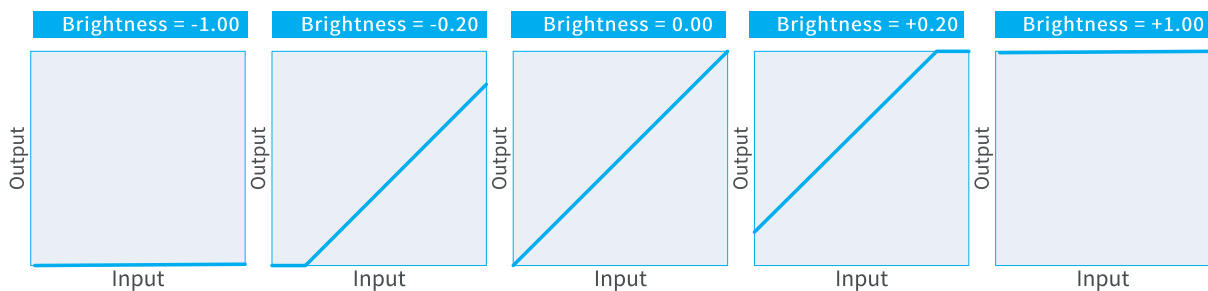
The `Brightness` parameter exclusively applies to the Response Control lookup table definition method.

It implements a control similar to the brightness control of a television monitor.

Brightness	Note
-1.00	Min. <code>Brightness</code> value: darkest output. The whole input range data gets transformed into the full black. This rule applies for any chosen <code>Contrast</code> value.
0.00	Default <code>Brightness</code> value.



Brightness	Note
	<p>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.</p> <p>Any increase in the brightness towards +1.00 results into a lighter output.</p> <p>Any decrease of the brightness towards -1.00 results into a darker output.</p>
+1.00	<p>Max. <code>Brightness</code> value: lightest output.</p> <p>The whole input range data gets transformed into the full white. This rule applies for any chosen <code>Contrast</code> value.</p>



**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:

$$\text{Gain} = 10^{2 \times (\text{Contrast} - 1)}$$

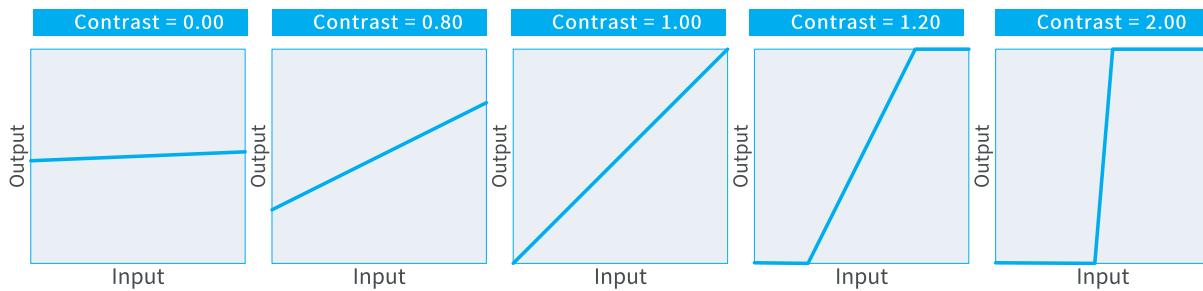
Contrast	Gain	Note
0.00	0.01	Min. <code>Contrast</code> value; smallest gain
1.00	1	Default <code>Contrast</code> value; unity gain
2.00	100	Max. <code>Contrast</code> value; largest gain

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):

$$\text{Contrast} = 1 + \text{Gain}(\text{dB}) / 40$$



**Effect of Contrast when all other controls are set to their default value:**  
 Brightness = 0.00; Visibility = 0.00; Negative = FALSE

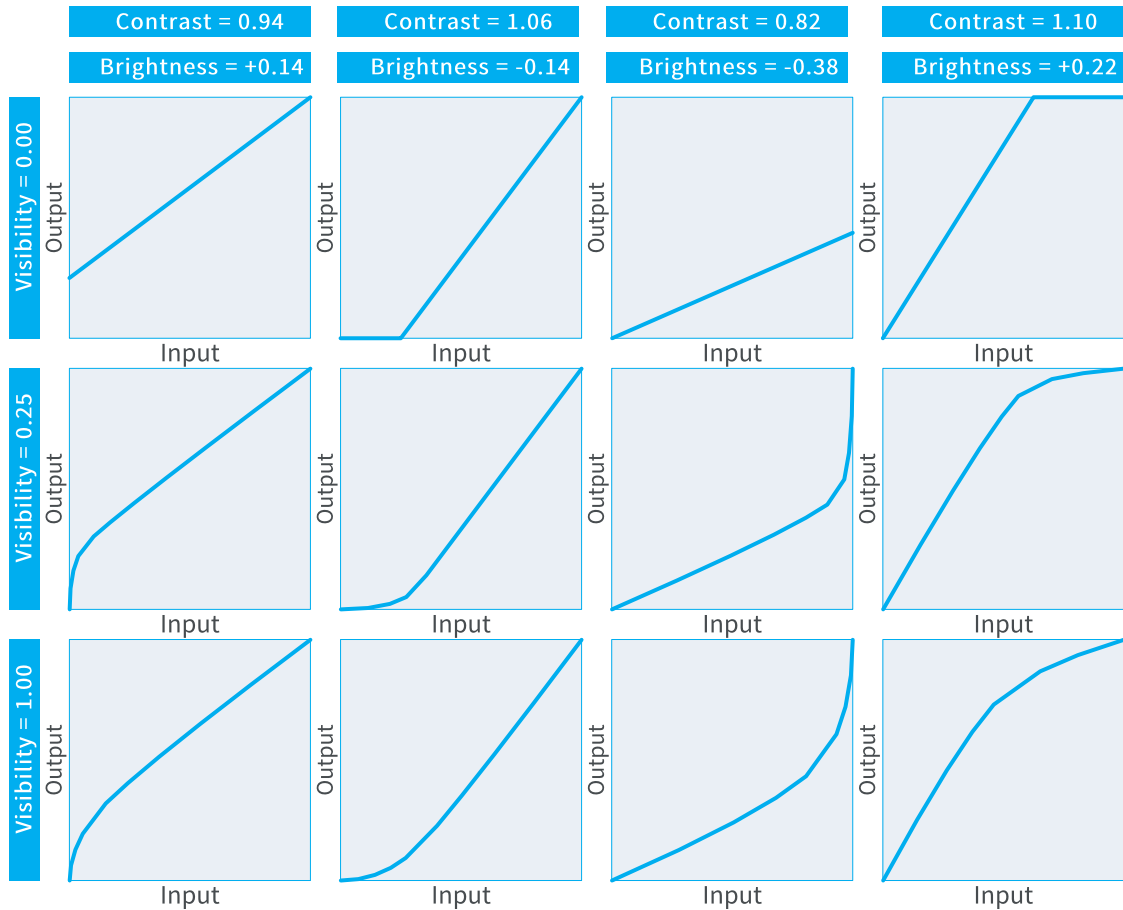
### Visibility Parameter

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 <code>Visibility</code> value. This generates the piecewise linear transformation curves. Choosing values closer to +1 generates smoother curves.
1.00	100	Max. <code>Visibility</code> value.



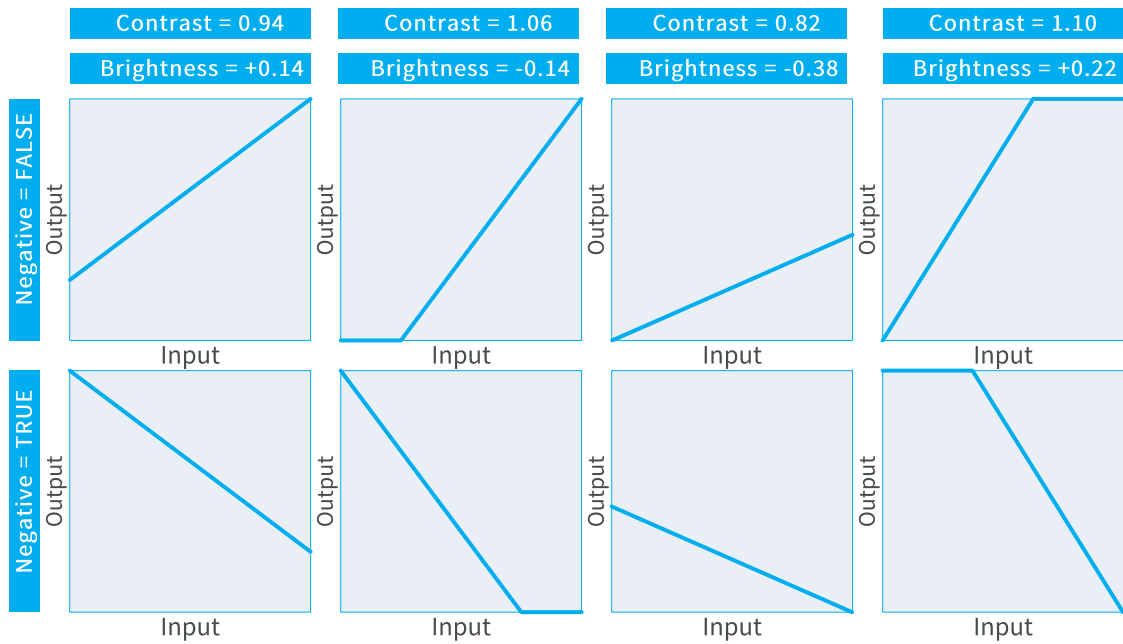
**Effect of Visibility for typical values of Contrast and Brightness parameters assuming that Negative = FALSE**

### 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:

$$\text{Output} = \text{Input}^\gamma$$

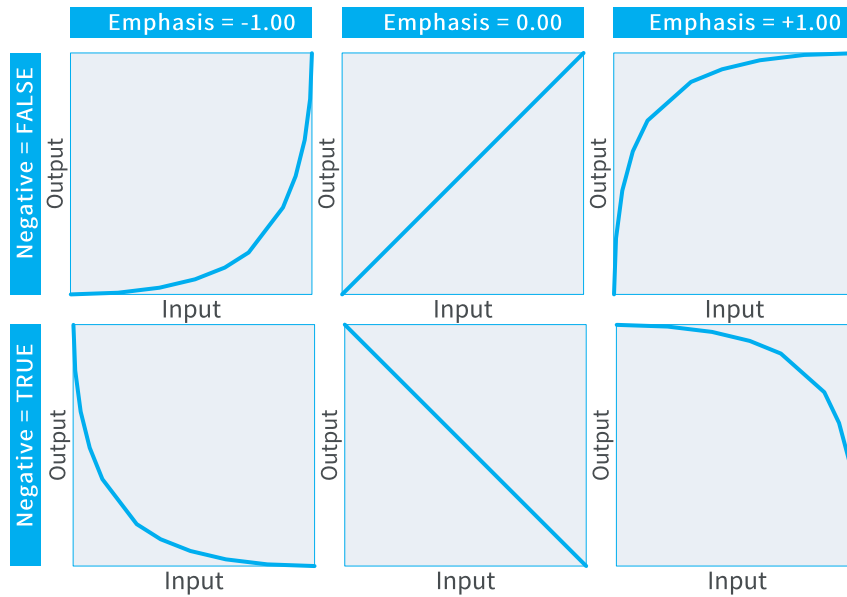
The  $\gamma$  - Gamma - exponent is mathematically linked to `Emphasis` by:

$$\gamma = 10^{-\text{Emphasis}}$$

Emphasis	Gamma	Note
1.00	0.1	Max. Emphasis value; smallest $\gamma$ value
0.00	1	Default Emphasis value; linear law
-1.00	10	Min. Emphasis value; largest $\gamma$ value

To achieve a required given  $\gamma$ , `Emphasis` should be set to:

$$\text{Emphasis} = -\log_{10}\gamma$$



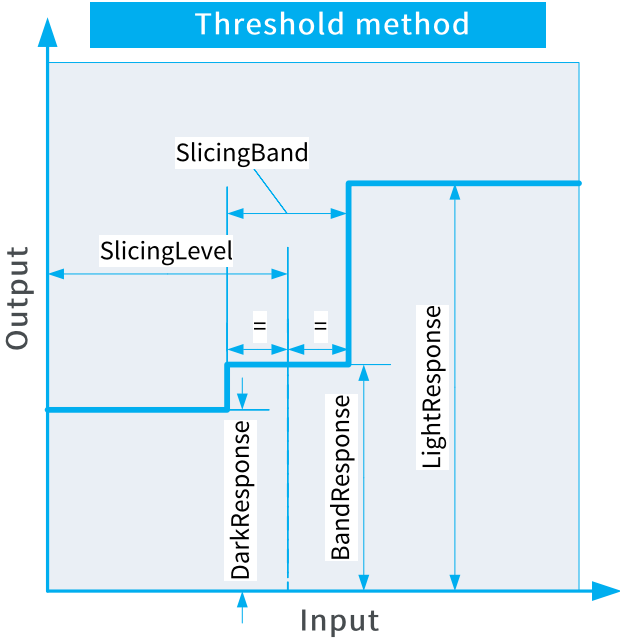
Emphasis effect for typical values of `Emphasis` and both values of `Negative`

### 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
Slicing Level	0.00	0.50	1.00
Slicing Band	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.

Refer to "[Monochrome Lookup Table Processing](#)" on page 167 for configurations applicable to monochrome pixels.

**Note:** *The lookup table configuration must be set prior to any other action.*

## Defining the lookup table content

---

Refer to "[LUT Content Definition](#)" on page 168 for a description of the parametric and tabular methods used for defining a lookup table content.

**Note:** *At least one lookup table set must be 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

Applies to: QuadG3 QuadG3DF Octo

## 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:

<b>Function</b>	<b>Mean(a,b)</b>
<i>Formula</i>	$(a + b + 1) \gg 1$

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:

<b>Function</b>	<b>Median(a,b,c,d)</b>
<i>Value 1 formula</i>	$\text{Min} [ \text{Max}(a,b) ; \text{Max}(c,d) ]$
<i>Value 2 formula</i>	$\text{Max} [ \text{Min}(a,b) ; \text{Min}(c,d) ]$

## Decoder operation

For each pixel of the source image, the CFA decoder computes two missing color components from surrounding pixels.

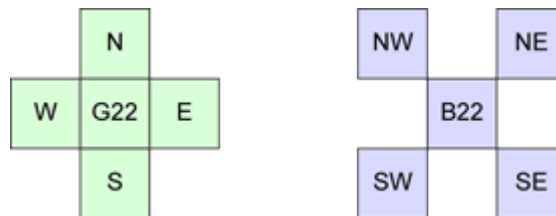
The text hereafter describes how the 4 central pixels located at positions 22, 32, 23 and 33 of a 4 x 4 Bayer CFA array are computed:

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

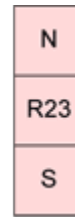
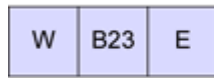
## Formulas for position 22



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

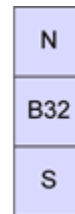
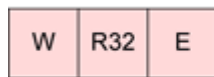
---



Component	Method	Formula
R23	Legacy, Advanced	Mean(N,S)
G23	Legacy, Advanced	G23
B23	Legacy, Advanced	Mean(W,E)

### Formulas for position 32

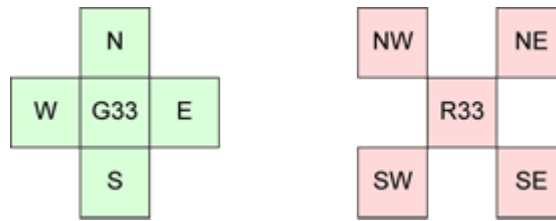
---



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:

Product	Firmware variant
1633 Coaxlink Quad G3	1-camera
1635 Coaxlink Quad G3 DF	1-camera
	1-df-camera
3602 Coaxlink Octo	1-camera
	2-camera

### 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.

**Important:** 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 62 control is inoperative, the Coaxlink card unpacks 10-bit, 12-bit or 14-bit pixels to lsb
- the "Pixel Component Re-Ordering" on page 64 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

Input Pixel Format	FFC	RedBlueSwap	Output Pixel Format
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	1-camera	1,000,000 pixels/s
1635 Coaxlink Quad G3 DF	1-df-camera	1,000,000 pixels/s
3602 Coaxlink Octo	1-camera	2,000,000 pixels/s
	2-camera	1,000,000 pixels/s per stream

**Note:** The peak pixel processing rate the Bayer CFA decoder approximately matches the performance of the DMA transfer when delivering RGB8 or BGR8 pixels.

**Note:** The peak pixel processing rate is significantly lower than the maximum aggregated input data rate achievable by the board! For instance, 2,500,000 vs. 1,000,000 pixels/s for a 1633 Coaxlink Quad G3.!

**Important:** 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!



**Important:** 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

---

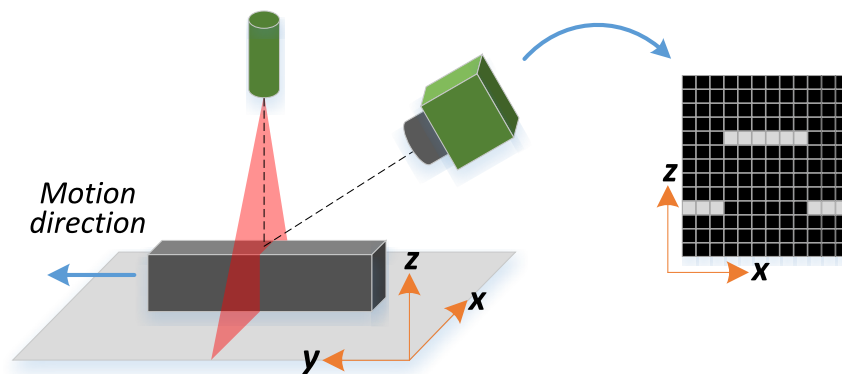
The hardware CFA decoder performs on-the-fly conversion with a negligible latency when the data throughput is NOT limited by the available PCI Express bandwidth!

## 11.7. Laser Line Extraction

Applies to: [Quad3DLLE](#)

## 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.



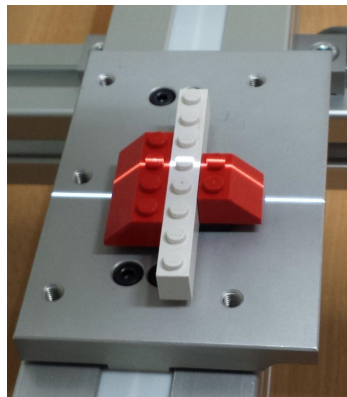
**Simplified laser triangulation system**

## Laser Line Extraction Algorithms

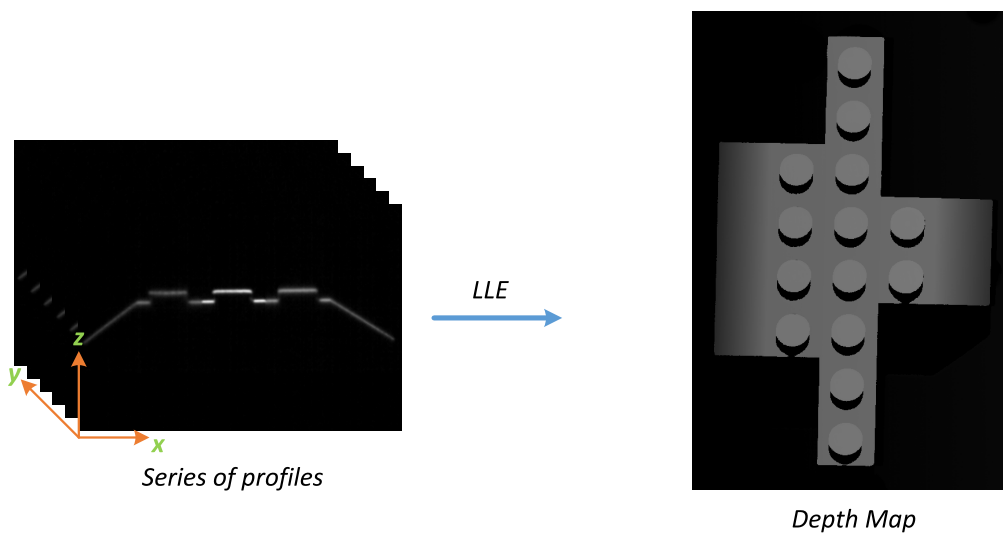
A Laser Line Extraction (LLE) algorithm is required to create a *depth map* based on a sequence of profiles of the object captured by the camera sensor.

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.



*Measured object*

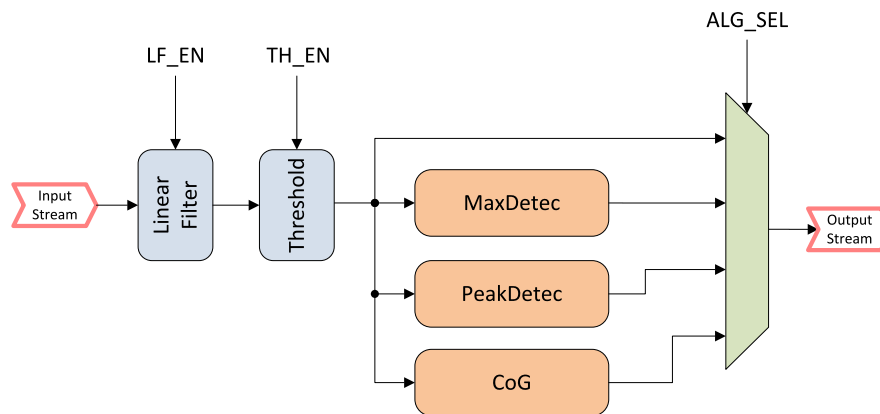


**Depth map generation**

## 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**

## LLC Processing Core Characteristics

### Absolute Maximum Input XSize

- All algorithms: 8192 pixels

### Max. Effective Input 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 ROI

### Maximum Performance

- All algorithms:
  - 2,500 megapixels/s
  - 4,700 profiles/s from a 2048 x 256 or 4096 x 128 image
  - 9,500 profiles/s from a 2048 x 128 image
  - 19,000 profiles/s from a 1024 x 128 image
  - 38,000 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.

### Available Output Types

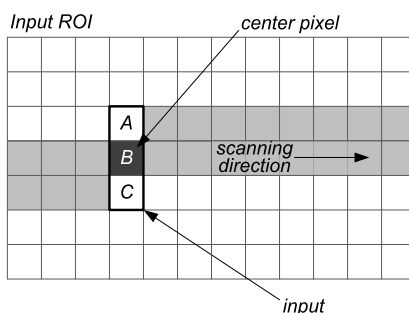
- All algorithms:
  - Depth map
  - Raw camera image

**Important:** Not both outputs simultaneously!

## Linear Filter

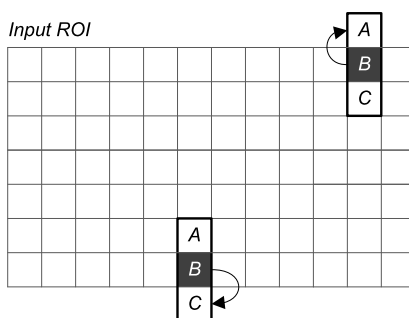
The *Linear Filter* module applies a convolution operator on a 1x3 sliding window directly on the data flow as it comes from the camera. The 3 elements of the convolution kernel (A, B, and C) are configurable accepting any positive integers where the sum of the 3 elements is a value between 1 and 512.

This figure illustrates the positioning of the convolution kernel elements within a given ROI:



### Linear Filter kernel disposition

When an element of the kernel is located outside of the ROI boundaries (typical sliding window problem), its input pixel value is replaced by the central window pixel.



### Linear Filter behavior at the ROI boundaries.

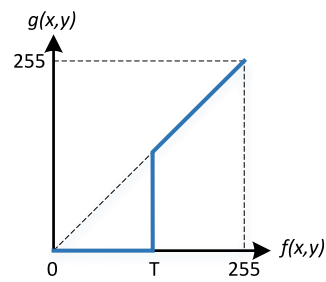


## Coring Threshold

The coring is a non-linear operator that performs a simple segmentation technique, classifying pixels into two categories according to the following rule:

$$g(x, y) = \begin{cases} f(x, y), & f(x, y) > T \\ 0, & \text{otherwise} \end{cases}$$

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 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 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 ROI height.
- No MAX intensity is detected in an 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 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 ROI YSize supported by the Peak Detection module is almost 256 Pixels.

When more than one peak is detected on a given 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 ROI height supported.
- No line is detected in an 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 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 ROI YSize supported by the Center of Gravity module is almost 256 Pixels.

When more than one peak is detected on a given 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:

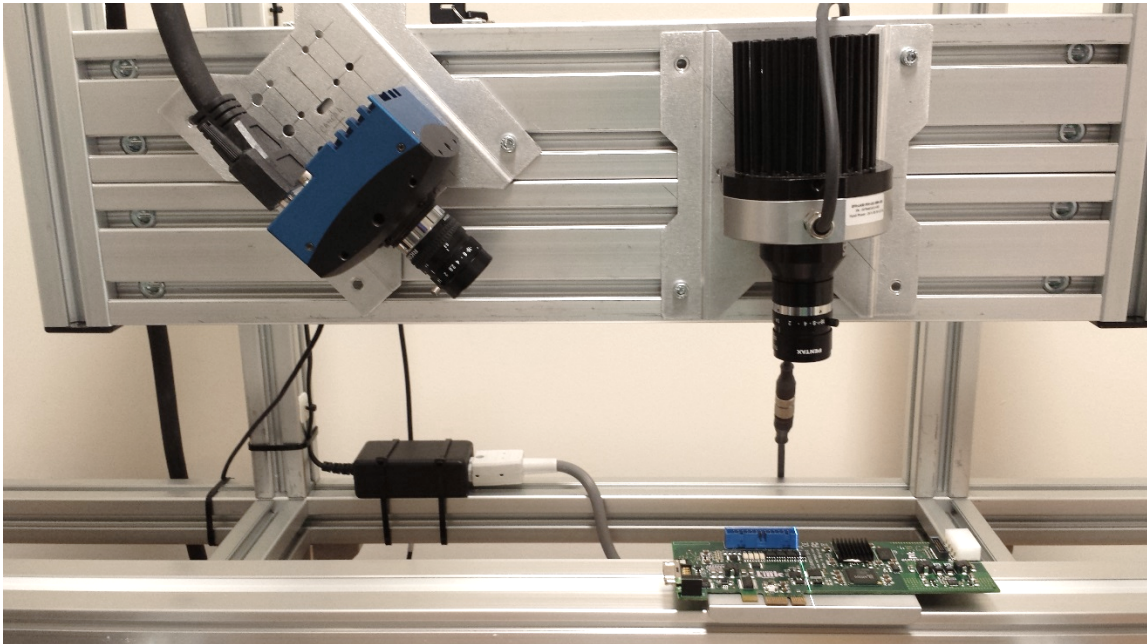
- The detected line position is above the max ROI height supported.
- An overflow occurs in an internal Sum. This condition can occur when in a same ROI column a large number of *successive* pixels present intensities above the threshold level.
- No line is detected in an ROI column.

# 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:

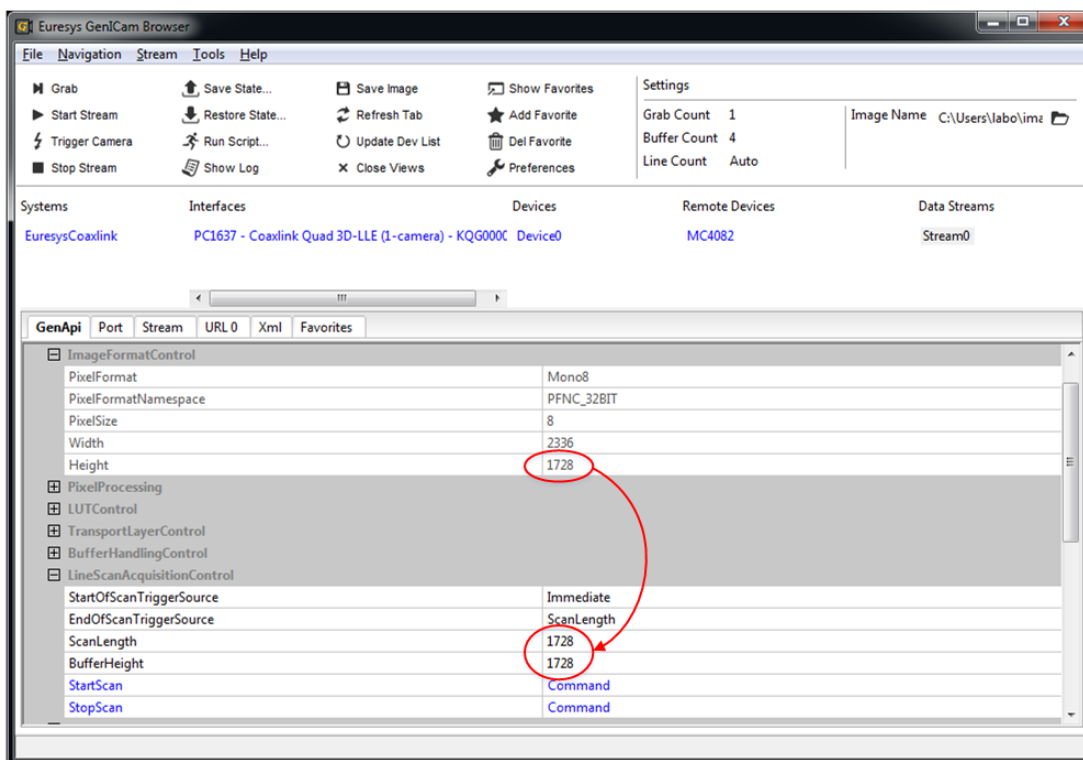


**Laser line triangulation setup**

## Defining the ROI

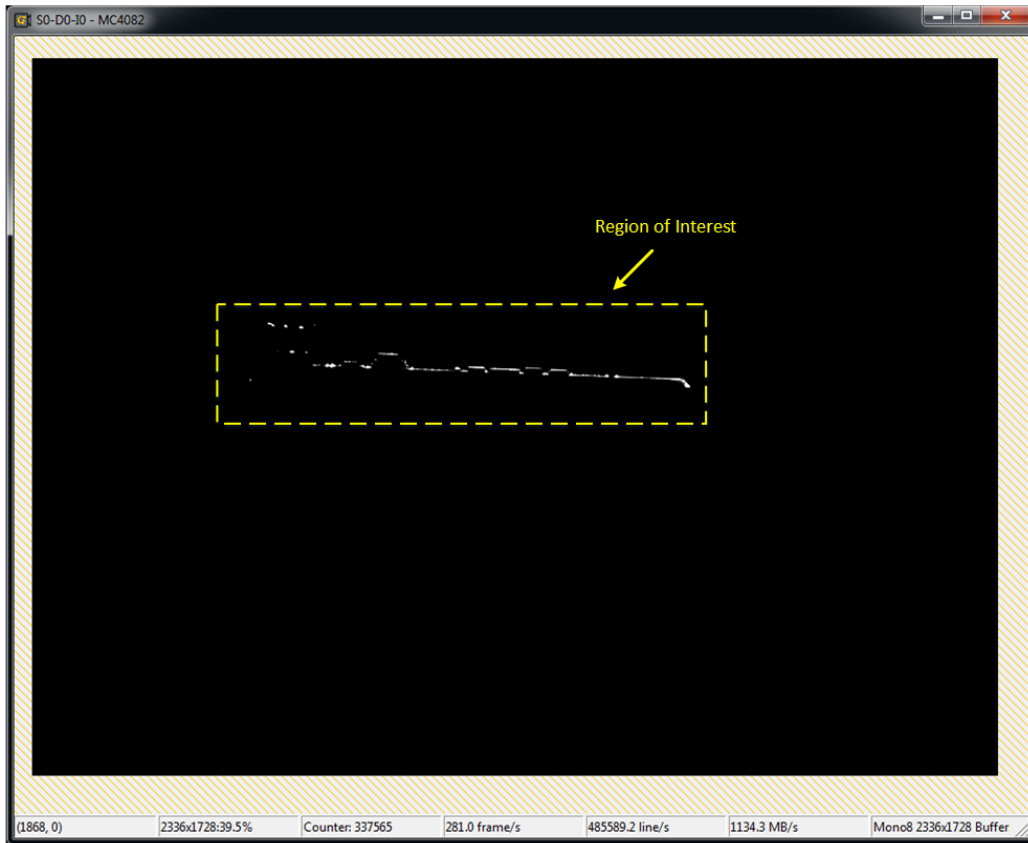
The defined ROI should cover the whole sensor region where the laser line variations occurs. The initial step is to set a full resolution ROI to verify the overall image.

To create a full resolution ROI, the ScanLength and BufferHeight must be set to the same Height value set on the “Remote Device” (camera). All these parameters can be seen in the in the “GenApi” tab of the “Data Streams” module as shown hereafter:

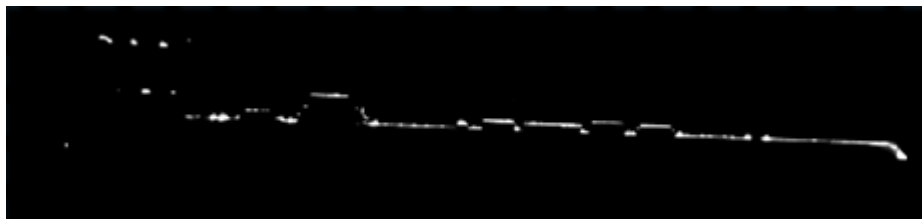


**Full resolution ROI**

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)**

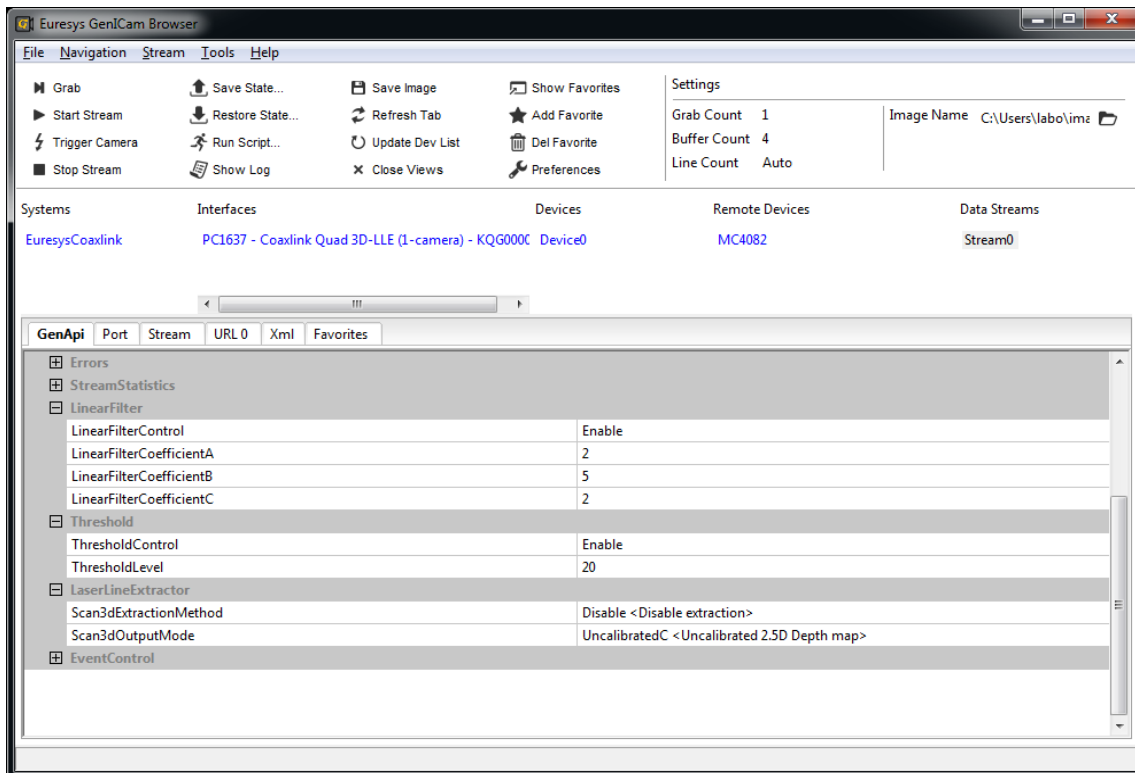


**ROI Expanded View**



## Setting Filtering and Thresholding

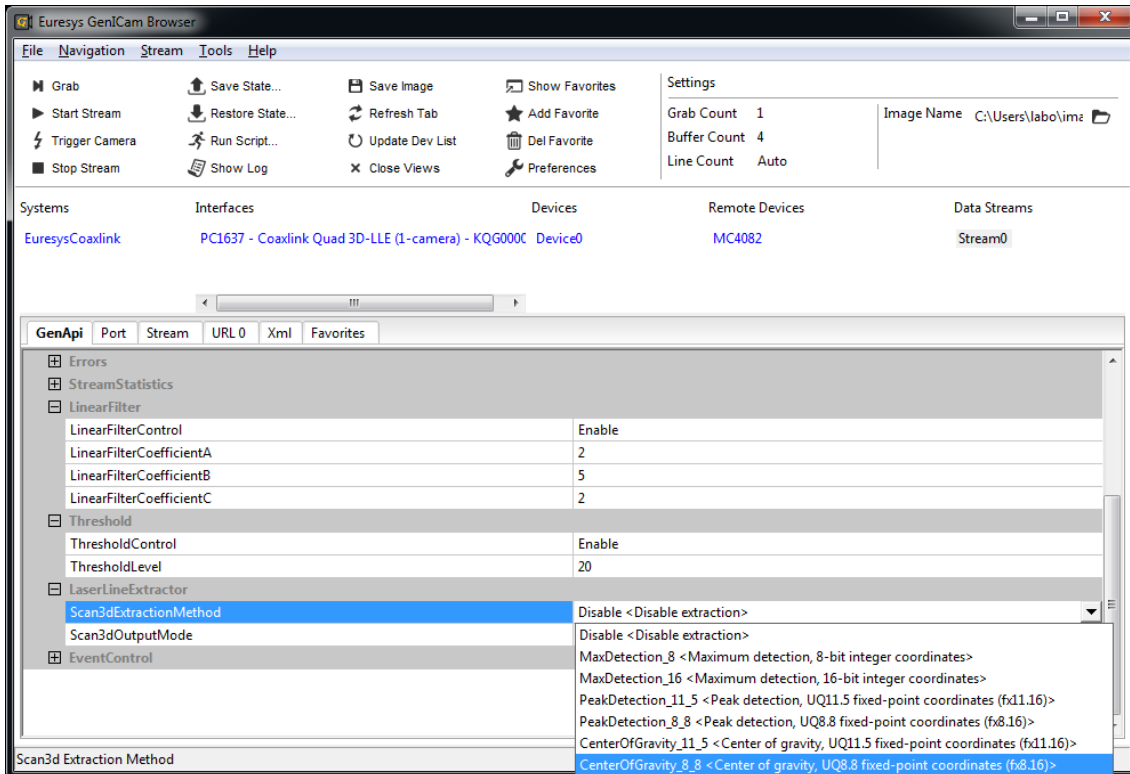
It is possible to activate a pre-processing stage in the LLE process. In the example shown below, the threshold level is set to 20 and the convolution filter kernel is set with  $A=2$ ,  $B=5$ , and  $C=2$ . With these parameters, we will have a smoother image where any background noise below the threshold 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.

#### ROI YSize

The number of effective vertical pixels in a 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 input ROIs.
- `PeakDetection`: The two versions of the `PeakDetection` module offer a trade-off between sub-pixel resolution and maximum effective 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 ROI size. The `CenterOfGravity` was designed to be robust to noisy inputs and wide lines.

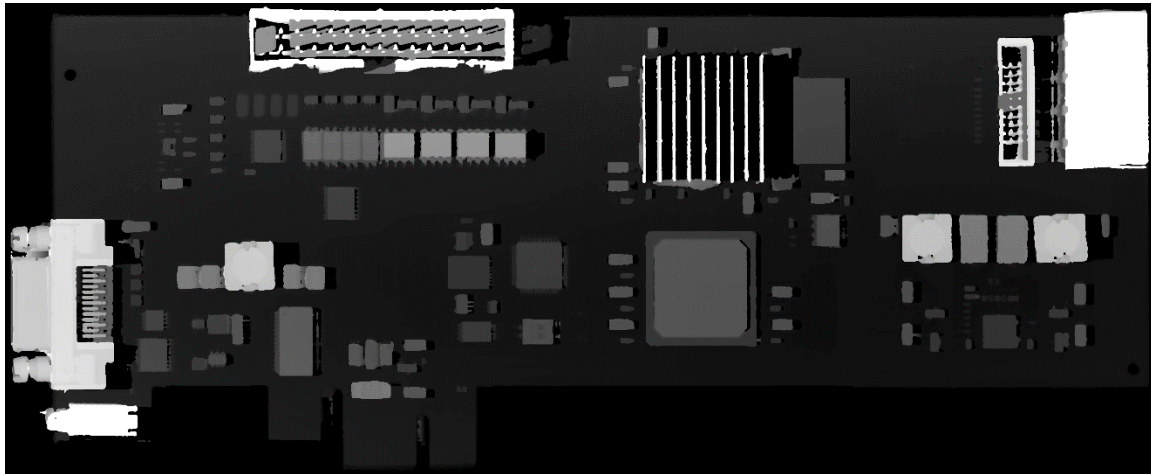
## Defining Scan Length and Buffer Size

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.

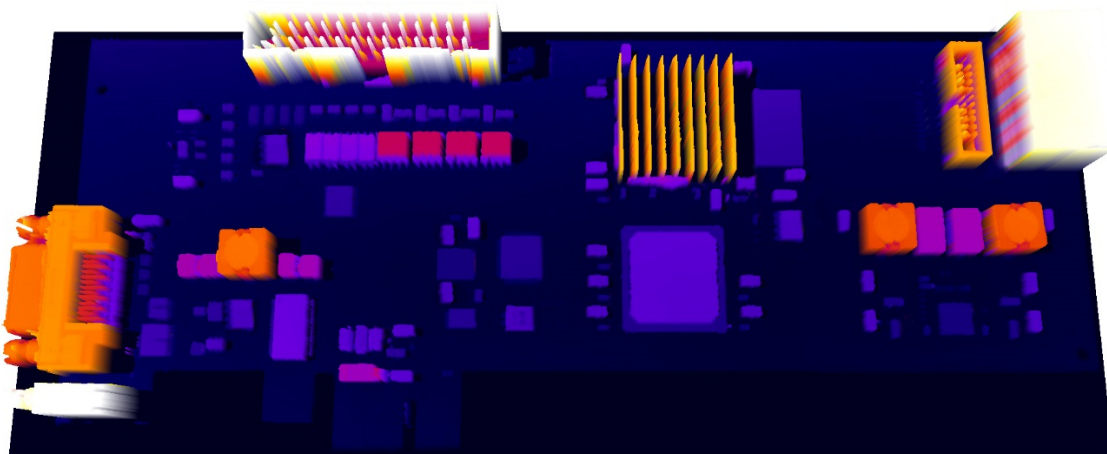
### Scanning Results

---

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 UQ<sub>m.n</sub> system is a representation of fixed-point numbers in the Q-format where Q designates a number in the Q-format notation, U preceding Q indicates an unsigned value, m indicates the number of bits of the integer portion of the number, and n designates the fraction portion of the number. For instance, UQ<sub>11.5</sub> is an unsigned fixed-point value represented by 16-bit words with 11 integer bits and 5 fractional bits. Other valid notations for UQ<sub>11.5</sub> are fx<sub>11.16</sub> and 0:11:5.

## How to convert “UQ<sub>11.5</sub>” fixed-point values to floating-point

---

The UQ<sub>11.5</sub> 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 “UQ<sub>8.8</sub>” fixed-point values to floating-point

---

The UQ<sub>8.8</sub> fixed-point values produced by the laser line extractor are 16-bit words representing values with 8 integer bits and 8 fractional bits.

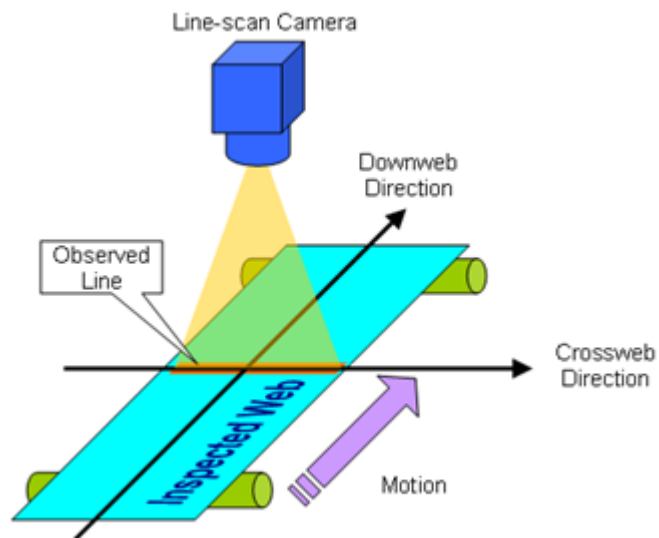
To convert from this format to float, first convert the value to float, then divide by  $2^8 = 256$ .

```
float toFloat(unsigned short x) {  
    float f = x;  
    return f / 256;  
}
```

## 11.8. Line Scan Acquisition

# Line Scan Acquisition Principles

## Line Scan Imaging



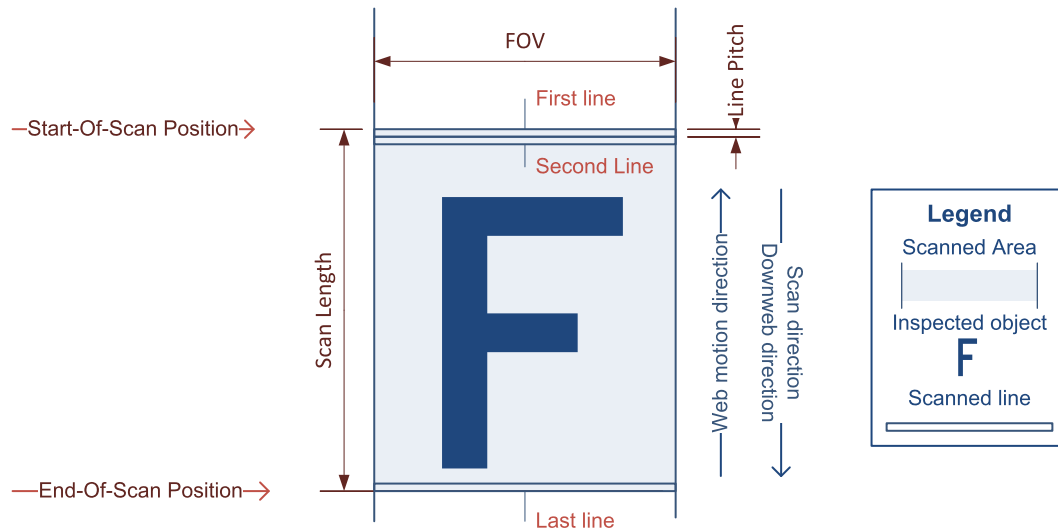
Typical line-scan imaging system

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	<p><b>Variable Camera cycle.</b></p> <p>The <b>web speed is variable</b> and the <b>camera cycle rate is kept proportional to the web speed.</b></p> <p>The frame grabber builds the object image by capturing all the successive lines delivered by the camera.</p>
CCC	<p><b>Constant Camera Cycle.</b></p> <p>The camera operates at a constant cycle rate and the frame grabber captures all the successive lines delivered by the camera.</p>

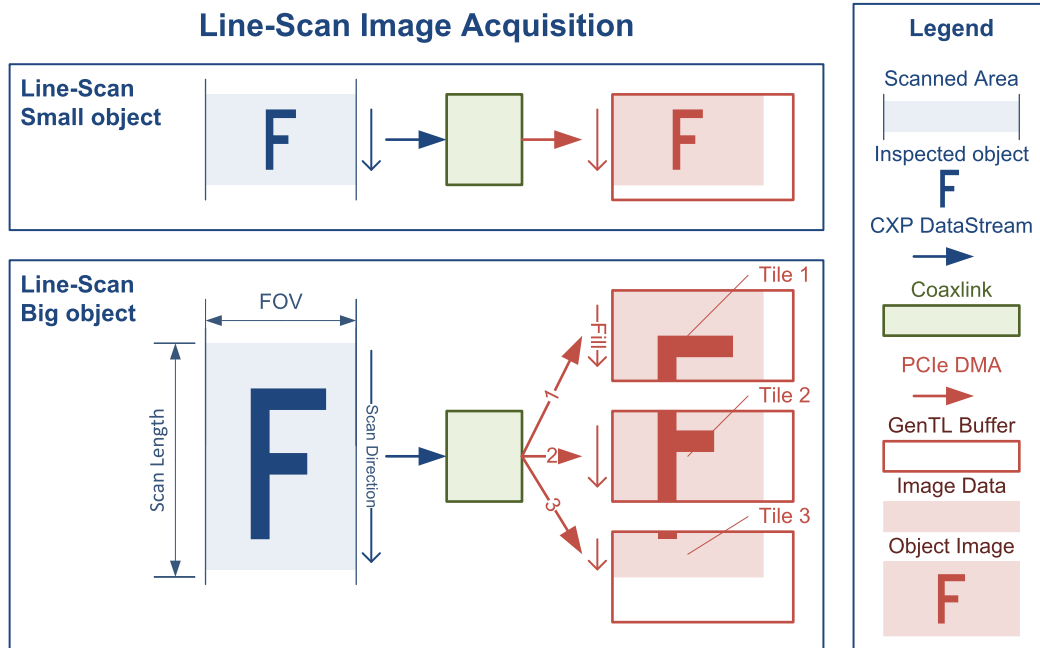
The VCR method requires:

- 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

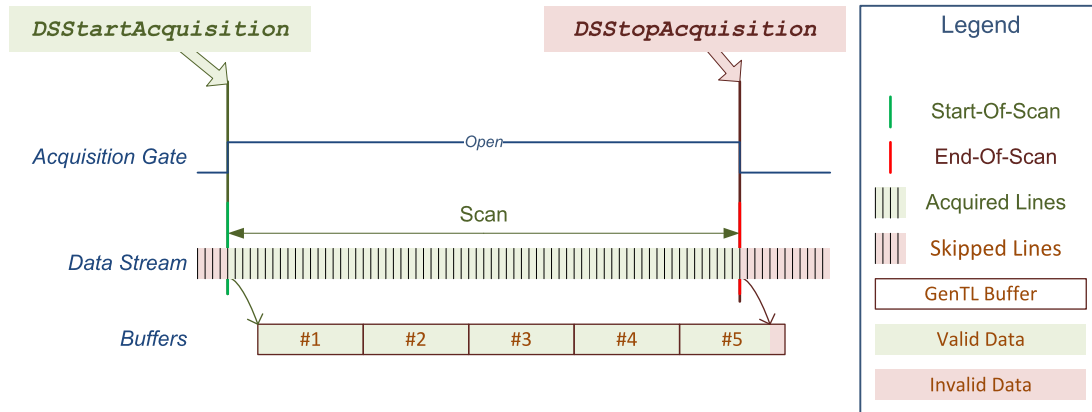
In line-scan imaging, GenTL buffers are filled according to the following rules:

- 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.

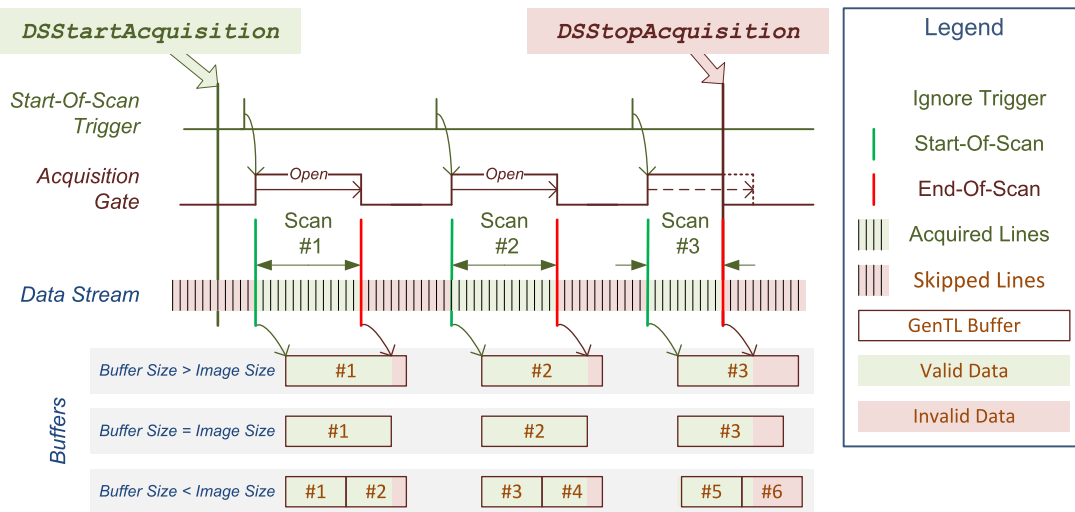
When the `DSStartAcquisition` function is called, the scanning starts at the next line boundary.

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 = 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`.

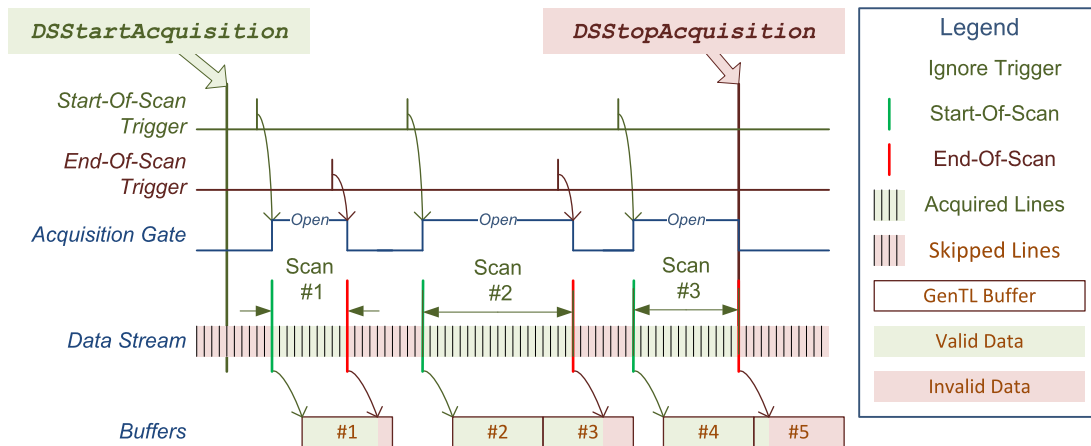
The acquisition controller ignores any Start-of-Scan trigger event while a scanning is in progress.

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.

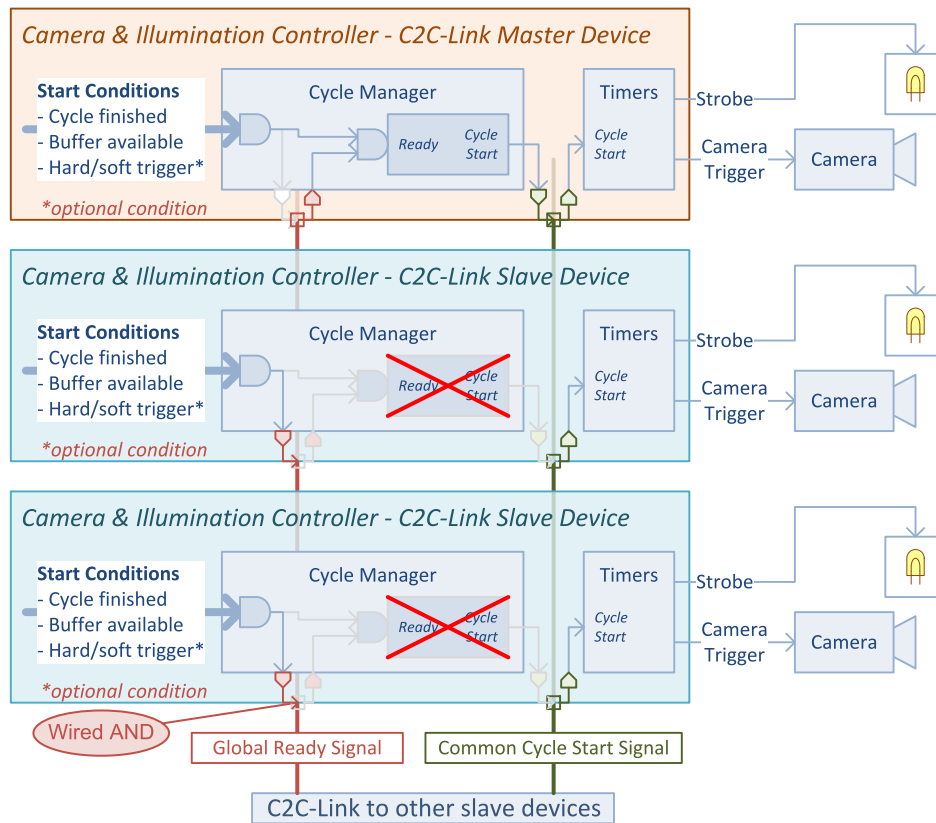
## 11.9. CIC Synchronization



## CIC Synchronization Principle

The synchronization is achieved by synchronizing the **CIC Cycle Trigger** of all involved CIC 's using the **C2C-Link** interconnect.

A C2C-Link interconnects two or more Camera Illumination Controllers. One CIC is configured as the **C2C-Link Master**, the others are configured as a **C2C-Link Slave(s)**.



### Configuration with one C2C-Link Master (top) and two C2C-Link Slaves:

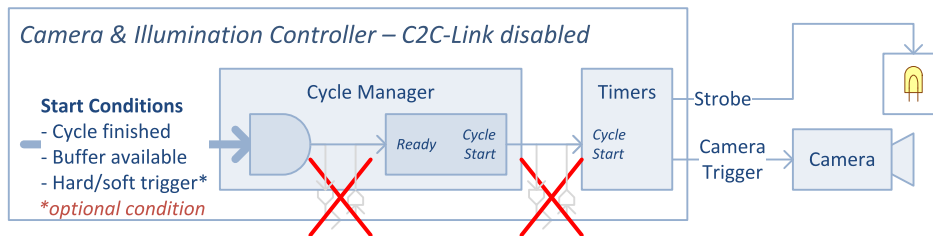
The C2C-Link Master:

- Generates a Common **Cycle Start** event when all the start conditions of all the synchronized CIC's, including itself, are satisfied.
- Delivers a common **Cycle Start** signal to all the synchronized CIC's.
- Upon reception of the Common **Cycle Start**, generates a **Strobe** pulse and a **Camera Trigger** pulse according to its timers settings.

A C2C-Link Slave:

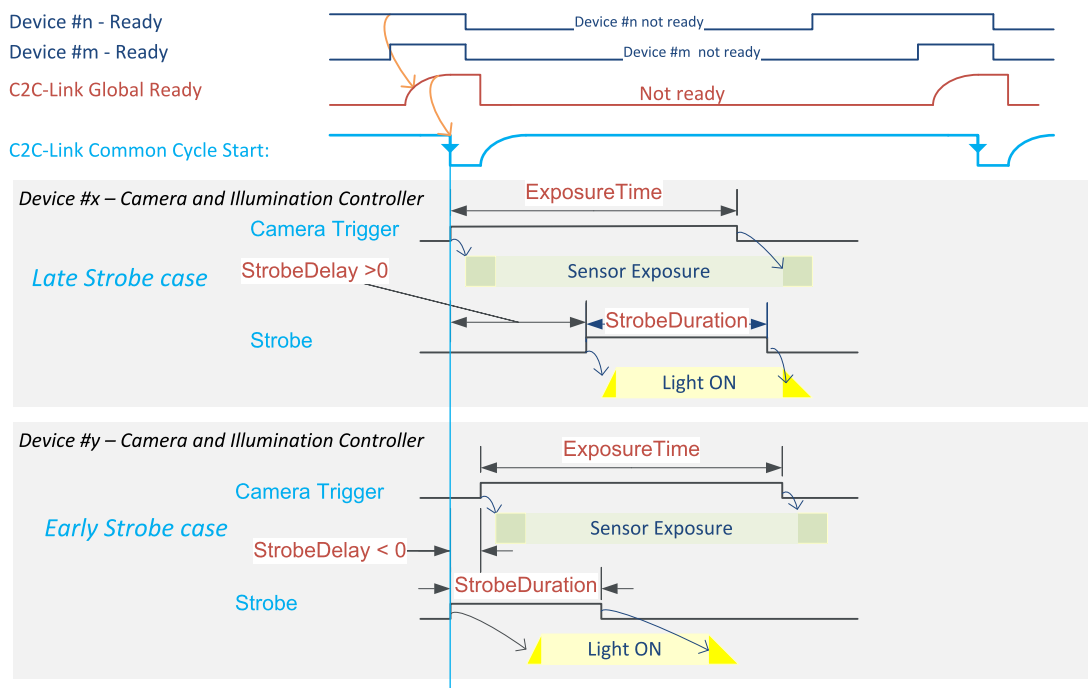
- Forces the **Global Ready** signal to 'false' until its start conditions are all satisfied
- Upon reception of the Common **Cycle Start**, generates a **Strobe** pulse and a **Camera Trigger** pulse according to its timers settings.

The C2C-link is kept disabled for devices that don't need to be synchronized.



**CIC device with disabled C2C-Link**

# CIC Synchronization Timing Diagram



**CIC Synchronization through C2C-Link timing diagram**

The above diagram shows the timing diagram of two consecutive common **Cycle Start** events:

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.

## 11.10. C2C-Link

## C2C-Link Description

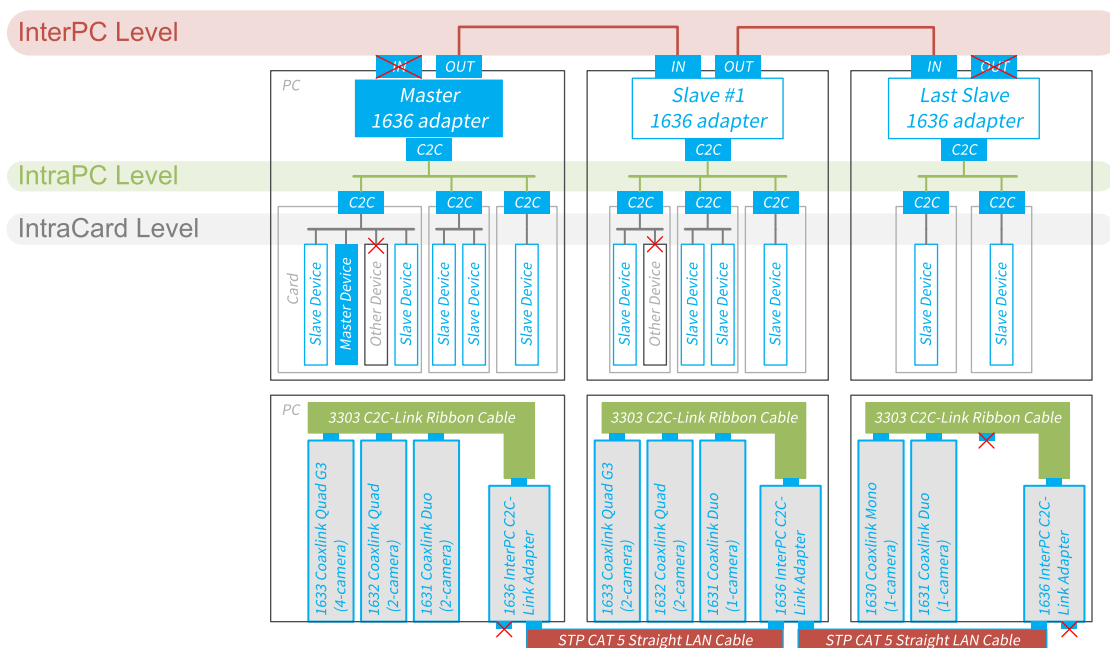
The C2C-Link is a hardware communication medium allowing a single “**C2C-Link master device**” to **reliably** share 1 or 2 trigger events with multiple “**C2C-Link slave devices**”.

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.

The C2C-Link feedback channel allows any C2C-Link device to indicate to the C2C-Link master device that it is not able to accept a trigger.

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 Configurations

C2C-Link Configuration	Interconnection Levels
IntraCard	IntraCard only
IntraPC	IntraPC + IntraCard (optional)
InterPC	InterPC + IntraPC+ IntraCard (optional)

# C2C-Link 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	$\mu$ s
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  $\times$  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
<b>Total Trigger Delay</b>	<b>0</b>	<b>5</b>	<b>10</b>	<b>ns</b>

### 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
<b>Total Trigger Delay for devices of the Master InterPC segment</b>	<b>0</b>	<b>5</b>	<b>10</b>	<b>ns</b>

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

Parameter	Min.	Typ.	Max.	Units
(3) Slave1 1636 InterPC C2C-Link Adapter – InterPC-to-IntraPC delay		20		ns
(1) Slave1 IntraPC interconnection	0	5	10	ns
<b>Total Trigger Delay for devices of the Slave1 InterPC segment</b>		<b>145</b>		<b>ns</b>

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
<b>Total Trigger Delay for devices of the Slave2 InterPC segment</b>		<b>245</b>		<b>ns</b>

## C2C-Link Use Cases

### **CIC Synchronization**

---

The Cycle Trigger event of the Camera and Illumination Controller of the "C2C-Link Master Device" is distributed to one or more "C2C-Link Slave Devices".

The CIC Cycles of all the devices start simultaneously. However, the Cycle Timing parameters can be configured individually.

Refer to "[CIC Synchronization](#)" on page 216 for more information.

## C2C-Link Availability

One C2C-Link connector is available on all Coaxlink products for extension of the C2C-Link beyond the card boundary.

One **IntraCard C2C-Link** interconnect is available only in multi-device firmware variants, namely:

- 2-camera
- 2-camera, line-scan
- 4-camera,
- 4-camera, line-scan

**CIC Synchronization** is available on all firmware variants except:

- 1-df-camera
- 1-df-camera, line-scan

# C2C-Link Setup

## General Procedure

---

- 1. C2C-Link hardware setup:** This step is specific to each C2C-Link Configuration. Refer to sections hereafter.
- 2. GenAPI setup:** Assign a value to the `C2CLinkConfiguration` GenTL Device feature.

## IntraCard C2C-Link Configuration Hardware Setup

---

This configuration exclusively uses FPGA internal resources to build the C2C-Link interconnect; it doesn't require any additional hardware!

## IntraPC C2C-Link Configuration Hardware Setup

---

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.

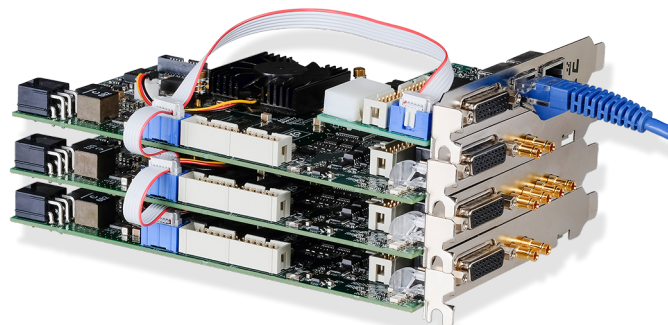
## InterPC C2C-Link Configuration Hardware Setup

---

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. Refer to "[Adapter Powering](#)" on page 237.
- 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. Refer to "[InterPC Interconnect](#)" on page 237.

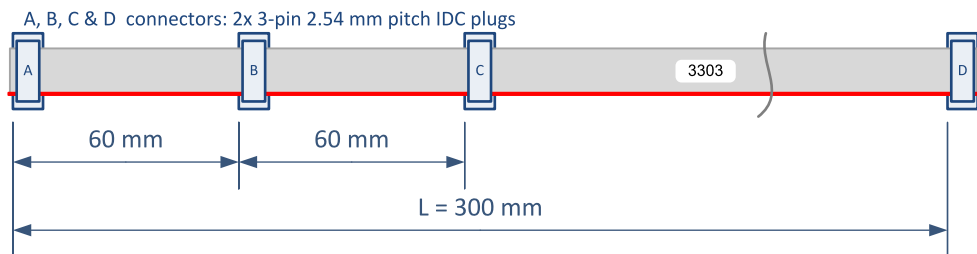


**IntraPC segment of an InterPC configuration**

## 3303 C2C-Link Ribbon Cable

3303 C2C-Link Ribbon Cable is an accessory product used for Intra-PC C2C-Link interconnection.

### 3303 C2C-Link Ribbon Cable



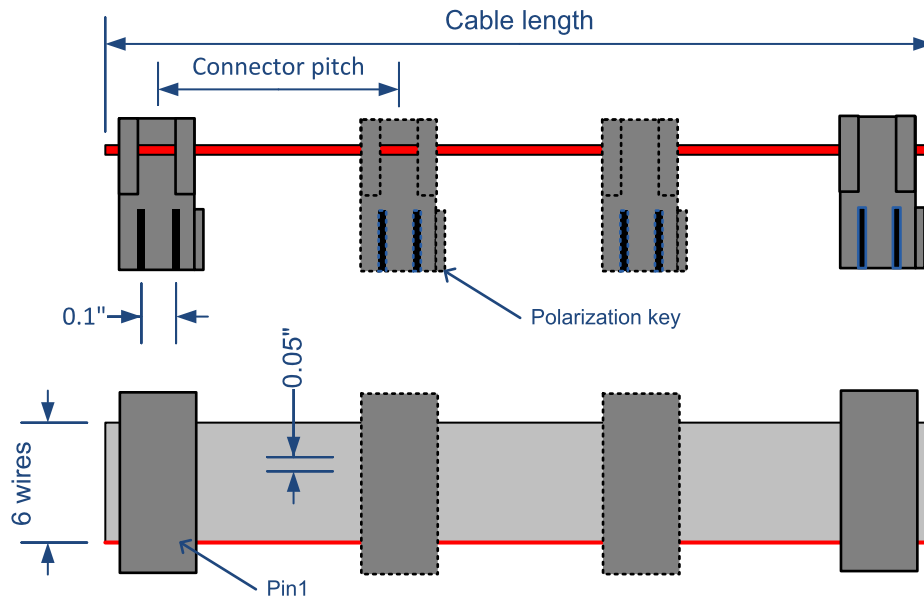
### 3303 C2C-Link Ribbon Cable assembly

The 3303 C2C-Link Ribbon Cable is a 6-conductor 0.05-in pitch ribbon fitted with 4 6-pin female ribbon cable connectors.

This cable is used for interconnecting the C2C-Link connectors of up to 4 cards located in the same PC.

## Custom C2C-Link Ribbon Cable Assembly

Assembly instructions of a custom-made IntraPC C2C-Link interconnection.



**Custom C2C-Link Ribbon Cable Assembly**

The cable assembly is composed with:

- A piece of a 6-conductor 0.05-in pitch ribbon cable. For instance: *Belden's (9L280XX Series)*.
- Two or more pieces of a 2 x 3-pin female ribbon cable connectors. For instance: *TE connectivity 1-1658528-1*.

The cable assembly has:

- A maximum of 4 connectors allowing up to 4 cards to share the same C2C-Link.
- A maximum length of 60 cm.

**Note:** *The connector pitch(es) must be determined according to the actual card to card spacing in the Host PC.*

## 1636 InterPC C2C-Link Adapter

The 1636 InterPC C2C-Link Adapter is an accessory product for use as an **InterPC C2C-Link extender** and/or as a **HD26F I/O adapter**.

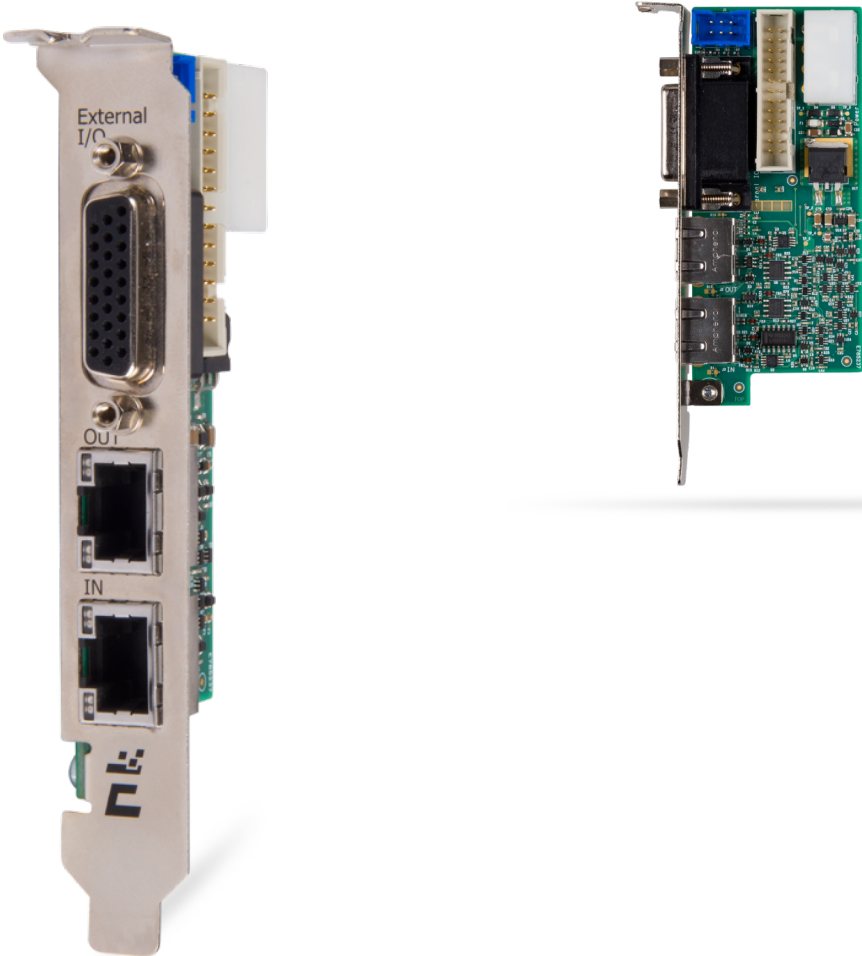
For a 1636 InterPC C2C-Link Adapter hardware description, refer to "[Hardware Description](#)" on [page 234](#).

For a description of the C2C-Link extender usage, refer to "[Using 1636 as C2C-Link Extender](#)" on [page 237](#).

For a description of the HD26F I/O adapter usage, refer to "[Using 1636 as HD26F I/O Adapter](#)" on [page 236](#).



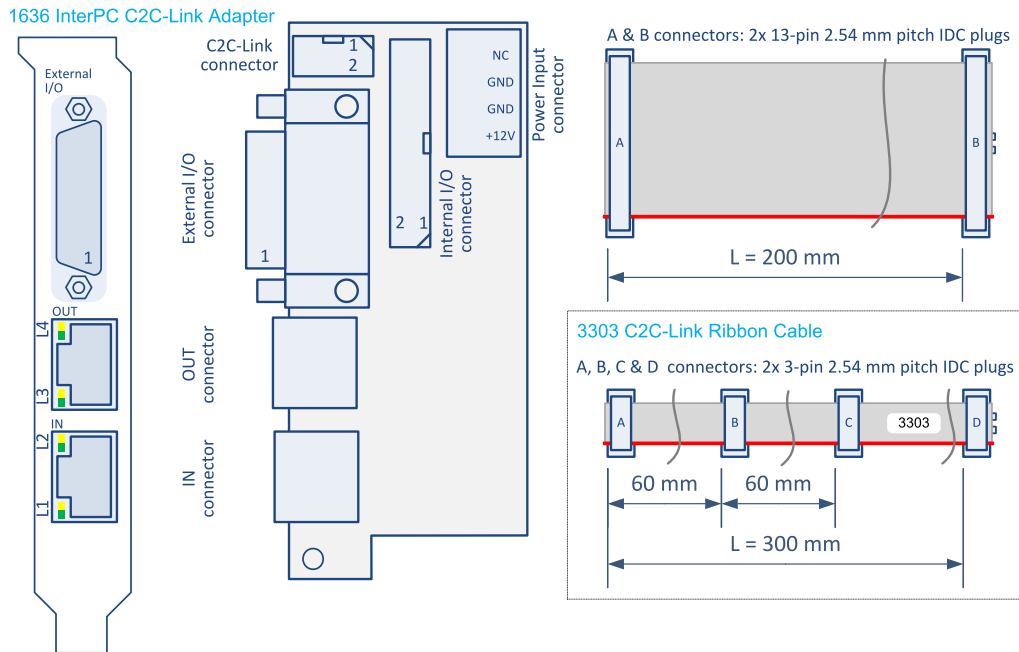
# Product Pictures



Pictures of 1636 InterPC C2C-Link Adapter

## Hardware Description

### Layout



### 1636 InterPC C2C-Link Adapter

The 1636 InterPC C2C-Link Adapter product accessory is composed of:

- A printed circuit board assembly fitted with a standard-profile PC bracket.
- A 200-mm 26-way ribbon cable.
- A 3303 C2C-Link Ribbon Cable.

### Connectors

The **External I/O connector** is a HD26F – 26-pin 3-row high-density female – Sub-D connector fitted on the bracket with UNC 4-40 screws. Refer to ["Using 1636 as HD26F I/O Adapter" on page 236](#) for the usage description and the pin assignments.

The **IN connector** and the **OUT connector** are RJ-45 8-pin sockets fitted on the bracket. Refer to ["Using 1636 as C2C-Link Extender" on page 237](#) for the usage description.

The **Internal I/O connector** is a 26-pin dual-row 0.1" pitch pin header with shrouding. Refer to ["Using 1636 as HD26F I/O Adapter" on page 236](#) for the usage description and the pin assignments.

The **C2C-Link connector** is a 6-pin dual-row 0.1" pitch pin header with shrouding. Refer to ["Using 1636 as C2C-Link Extender" on page 237](#) for the usage description.

The **Internal I/O connector** is a 26-pin dual-row 0.1" pitch pin header with shrouding. Refer to ["Using 1636 as HD26F I/O Adapter" on the next page](#) for the usage description.

The **Power Input connector** is a 0.2" pitch right-angled Disk Drive Power connector. Refer to ["Using 1636 as C2C-Link Extender" on page 237](#) for the usage description.

## Lamps

---

The **IN connector** and the **OUT connector** are each equipped with 2 green/yellow LED lamps named respectively **L1**, **L2**, **L3** and **L4**. Refer to ["Using 1636 as C2C-Link Extender" on page 237](#) topic for the usage description.

## Using 1636 as HD26F I/O Adapter

To use 1636 InterPC C2C-Link Adapter as an HD26F I/O adapter:

- Plug the A-connector of the supplied 200-mm 26-way ribbon cable to the **Internal I/O** connector of the 1636 InterPC C2C-Link Adapter
- Plug the B-connector to the **Internal I/O** connector of the target card.

**Note:** *No power supply connection is required when using the 1636 InterPC C2C-Link Adapter as an HD26F I/O adapter only.*

# Using 1636 as C2C-Link Extender

## Adapter Powering

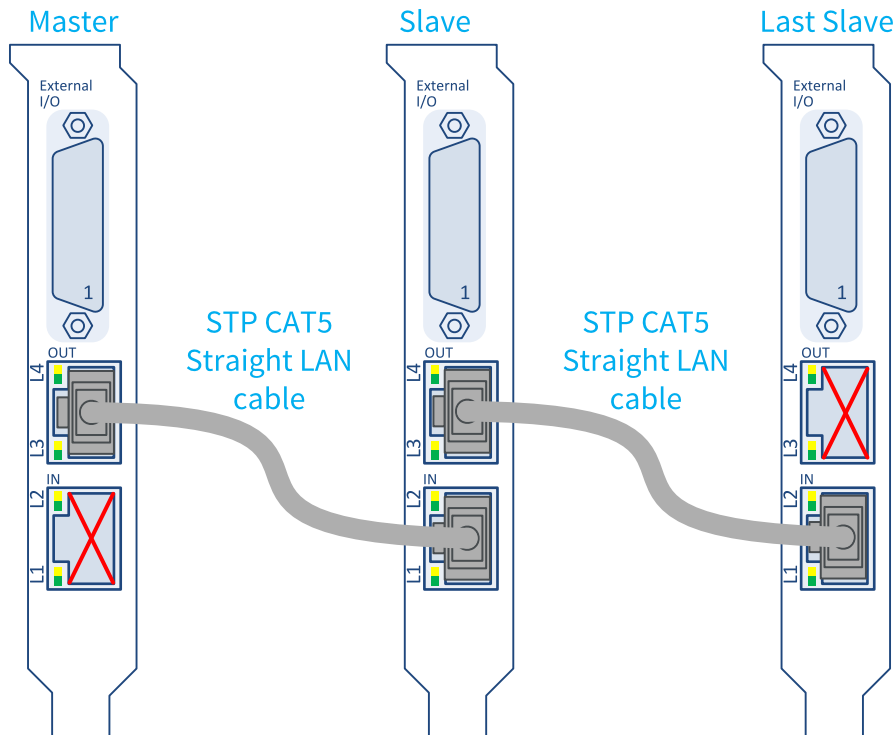
**Important:** The 1636 InterPC C2C-Link Adapter must be powered when it is used as a C2C-Link extender.

The user has two options to supply power to the adapter:

- From the Coaxlink card +12V power output through the 26-way ribbon cable attached to the **Internal I/O** connector.
- From the Host PC power supply through a Disk Drive Power connector cable plugged into the **Power Input** connector.

Parameter	Min.	Typ.	Max.	Units
+12 V DC Input voltage	11.0	12.0	13.0	V
+12 V Input power		1.8		W

## InterPC Interconnect



External wiring of a C2C-Link across 3 adapters.

The external wiring of the C2C-Link is made with RJ 45 CAT 5 STP straight LAN cables. N-1 cables are required to interconnect N adapters in a daisy-chain scheme.

The daisy-chain begins on the OUT connector of the Master adapter and ends at the IN connector of the Last Slave adapter.

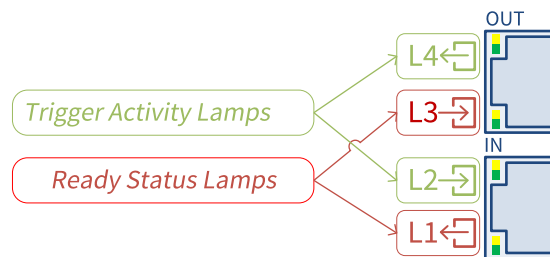
The IN connector of the Master adapter and the OUT connector of the Last Slave adapter are unused.

**Note:** *The adapter disables the signal drivers of the IN and OUT connectors to avoid electrical damages when it detects a bad or a missing connection.*

The InterPC cable drivers and receivers are not electrically isolated.

**Important:** *To avoid damages, the interconnected PCs must have a common ground reference.*

## Lamps



1636 InterPC C2C-Link Adapter lamps

### Trigger Activity Lamps

The L2 and L4 Lamps indicate the trigger activity on the LAN cable. L2 shows the activity on the received trigger signals; L4 shows the activity on the transmitted trigger signals.

Lamp State	Indication
Off	The LAN cable is unplugged or the adapter is not powered.
Green	No trigger activity. <i>No trigger events in the past 10 milliseconds.</i>
Yellow	Trigger activity. <i>One or more trigger events in the past 10 milliseconds.</i>

### Ready Status Lamps

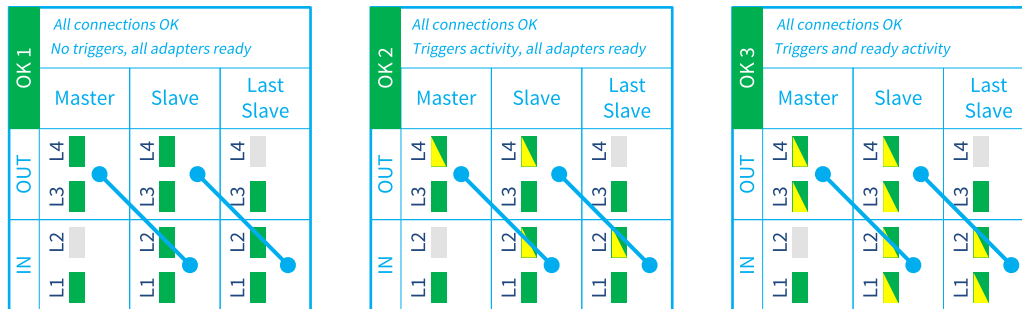
The L1 and L3 Lamps indicate the state of the ready signal on the LAN cable. L1 shows the state of the transmitted ready signal; L3 shows the state of the received ready signal.

Lamp State	Indication
Off	The adapter is not powered.
Green	Ready true.

Lamp State	Indication
	<p>For L1: all the C2C-Link devices attached to this adapter and the downwards adapters (if any) are ready.</p> <p>For L3: all the C2C-Link devices attached to the downwards adapters (if any) are ready.</p>
Yellow	<p>Ready false.</p> <p>For L1: one or more C2C-Link devices attached to this adapter and the downwards adapters (if any) are not ready.</p> <p>For L3: one or more C2C-Link devices attached to the downwards adapters (if any) are not ready.</p>

**Note:** Unlike the trigger activity lamps, the ready signals are not enlarged. Short-duration not-ready states are hardly visible!

### Adapters Array Lamp States - Normal Situations



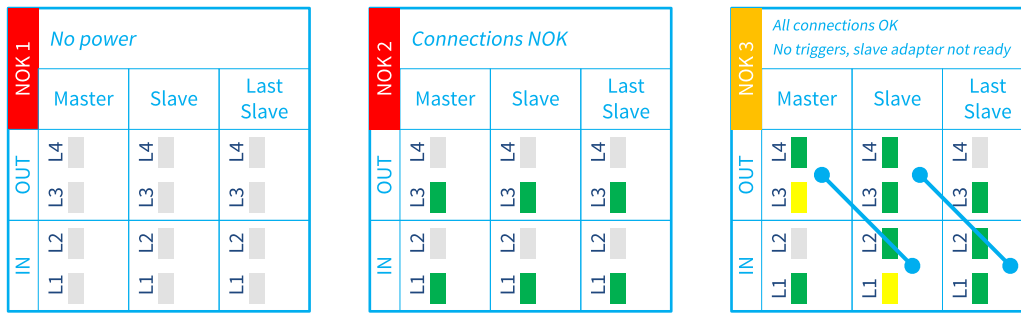
The above drawings show the lamps states of 3 daisy-chained adapters for 3 normal situations.

In the OK 1 situation, all adapters are ready to accept triggers but no triggers are sent by the master.

In the OK 2 situation, the master adapter sends triggers and the ready signal of all adapters is permanently high. The yellow/green toggling L2 and L4 lamps indicate the trigger activity. The steady green L1 and L3 lamps indicate that all adapters are permanently ready to receive triggers.

In the OK 3 situation, the master adapter sends triggers and the ready signal of all adapters is cycling. The yellow/green toggling L2 and L4 lamps indicate the trigger activity. The yellow/green toggling L1 and L3 lamps indicate that all adapters are not ready to receive triggers for a significant duration.

## Adapters Array Lamp States - Abnormal Situations



The above drawings show the lamps states of 3 daisy-chained adapters for 3 abnormal situations.

In the NOK 1 situation, no adapters are powered. All lamps are Off.

In the NOK 2 situation, all adapters are powered but all connections are missing or incorrect.

In the NOK 3 situation, all adapters are powered and all connections are OK, but the second adapter is not ready preventing the master to send new triggers. This situation is considered as abnormal when it persists.

## Troubleshooting Guide

Lamps state	Indication and possible causes	Action
All lamps Off	The adapter is not powered.	Apply power to the adapter
L2 Off L1 Green	The external connection to the IN connector is missing or incorrect.	For the master adapter, this is OK: nothing to do! For the other adapters: check and correct the connection to the OUT connector of the previous adapter in the daisy-chain.
L4 Off L3 Green	The external connection to the OUT connector is missing or incorrect.	For the last slave adapter of the daisy-chain, this is OK: nothing to do! For the other adapters: check and correct the connection to the IN connector of the next adapter in the daisy-chain.



## 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`.

**Important: Important:** *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::string encryptedKey=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.*