Contents

1 About this manual 2
   1.1 Composition of the manual 2
   1.2 How to get started? 2
      1.2.1 Introduction 2
      1.2.2 Basics 3
      1.2.3 Image acquisition concept 5
      1.2.4 Installation 5
      1.2.5 Programming 6

2 Imprint 7

3 Revisions 8

4 Graphic Symbols 10
   4.1 Notes, Warnings, Attentions 10
   4.2 Webcasts 10

5 Important information 11
   5.1 European Union Declaration of Conformity statement 11

6 Introduction 14
   6.1 What's inside and accessories 14

7 Quickstart 15
   7.1 Hardware installation 15
   7.2 Windows 15
      7.2.1 System Requirements 15
      7.2.2 Software installation 16
   7.3 Linux 20
      7.3.1 System Requirements 20
      7.3.2 Installing the mvIMPACT Acquire driver 21
   7.4 Connecting a camera 25
   7.5 Settings behavior during startup 25
8 Technical data
8.1 mvHYPERION-CLx .................................................. 28
  8.1.1 Block diagrams ............................................... 28
  8.1.2 Connectors ..................................................... 28
  8.1.3 Components ................................................... 37
  8.1.4 Device Feature And Property Lists ......................... 38
8.2 mvHYPERION-32R16 .................................................. 38
  8.2.1 Block diagram ................................................. 38
  8.2.2 Connectors ..................................................... 39
  8.2.3 Components ................................................... 40
8.3 mvHYPERION-HD-SDI .................................................. 41
  8.3.1 Block diagram ................................................. 41
  8.3.2 Connectors ..................................................... 42
  8.3.3 Components ................................................... 44
9 Application Usage ...................................................... 48
9.1 wxPropView .......................................................... 48
  9.1.1 How to work with wxPropView ............................... 48
  9.1.2 How to configure a device .................................... 70
  9.1.3 Command-line options ....................................... 94
9.2 mvDeviceConfigure ................................................... 95
  9.2.1 How to set the device ID ..................................... 95
  9.2.2 How to update the firmware .................................. 97
  9.2.3 How to recover a broken firmware update ................. 100
  9.2.4 How to allocate image memory ............................... 100
  9.2.5 How to disable CPU sleep states a.k.a. C states (< Windows 8) ................................................. 101
  9.2.6 Command-line options ....................................... 103
10 HRTC - Hardware Real-Time Controller ........................... 105
  10.1 Introduction ...................................................... 105
  10.1.1 Operating codes .............................................. 105
  10.1.2 Program controls ............................................. 105
  10.2 How to use the HRTC ........................................... 106
17 Use cases

17.1 scanCameras Working with line scan cameras .............................................. 121
17.2 Pass-through of digital input signals .............................................................. 121
17.3 Working with pulse start events ................................................................. 122
17.4 Working with an rotary encoder ................................................................. 123
   17.4.1 DigitalSignalA ......................................................................................... 124
   17.4.2 DigitalSignalB ......................................................................................... 124
   17.4.3 PulseMultiplication ............................................................................... 125
   17.4.4 Direction ................................................................................................. 125
   17.4.5 Mode ....................................................................................................... 125
   17.4.6 int Reset() ............................................................................................ 126
17.5 Working with a Basler Sprint line scan color camera .................................... 126
   17.5.1 Introduction ............................................................................................. 126
   17.5.2 RawLineAcquisition Mode ...................................................................... 128
   17.5.3 EnhancedRawLineAcquisition Mode ........................................................ 130
17.6 Working with trigger events .......................................................................... 134
   17.6.1 FrameStart .............................................................................................. 135
   17.6.2 FrameStart + FrameStop ......................................................................... 136
   17.6.3 AcquisitionStart ...................................................................................... 138
17.7 Synchronous acquisition with different camera settings ......................... 139
1 About this manual

1.1 Composition of the manual

The mvIMPACT Acquire manual for the MATRIX VISION frame grabbers is based on a modular concept. That means like in many object-oriented programming languages you have for each functionality your own “class”. Instead of classes, you have books. For example, if you want to know how images are acquired with the frame grabbers, have a look in the respective programming language chapter.

Here is a short summary about all books of the frame grabber manual:

- The manual starts with technical data of the frame grabber as well as a quick start chapter.

   Afterwards, you will find the different books:

   - **Application Usage** (p. 48)
     - The frame grabbers can also be managed via user interface. The program is called **wxPropView** (p. 48).

   - **DirectShow developers** (p. 114)
     - This is the documentation of the MATRIX VISION DirectShow_acquire interface.

   - **Use cases** (p. 121)
     - This book offers solutions and explanations for standard use cases.

**Note**

For C, C++, .NET developers, there are separate mvIMPACT Acquire manuals:

- "mvIMPACT_Acquire_API_CPP_manual.chm",
- "mvIMPACT_Acquire_API_C_manual.chm", and
- "mvIMPACT_Acquire_API_NET_manual.chm"

available as downloads from our website [http://www.matrix-vision.com](http://www.matrix-vision.com). The manuals contain chapter about

- how to link and build applications using mvIMPACT Acquire,
- how the log output for "mvIMPACT Acquire" devices is configured and how it works in general,
- how to create your own installer packages for Windows and Linux, and
- the general mvIMPACT Acquire API documentation.

1.2 How to get started?

1.2.1 Introduction

This chapter gives you a short overview, how to get started with a MATRIX VISION frame grabber and where to find the necessary information in the manual. It will also explain or link to the concepts behind the driver and the image acquisition. Furthermore it shows you how to get start programming own applications.
1.2 How to get started?

1.2.2 Basics

1.2.2.1 Driver concept

The driver supplied with the MATRIX VISION product represents the port between the programmer and the hardware. The driver concept of MATRIX VISION provides a standardized programming interface to all image processing products made by MATRIX VISION GmbH.

The advantage of this concept for the programmer is that a developed application runs without the need for any major modifications to the various image processing products made by MATRIX VISION GmbH. You can also incorporate new driver versions, which are available for download free of charge on our website.

The following diagram shows a schematic structure of the driver concept:

- 1 Part of any mvIMPACT Acquire driver installation package (Windows).
- 2 Separately available for 32 bit and 64 bit. Requires at least one installed driver package.
- 3 See 2, but requires an installed version of the mvBlueFOX driver.
- 4 Part of the NeuroCheck installer but requires at least one installed frame grabber driver.
- 5 Part of the mvIMPACT SDK installation. However, new designs should use the .NET libs that are now part of mvIMPACT Acquire ("mv.impact.acquire.dll"). The namespace "mv.impact.acquire" of "mv.impact.acquire.dll" provides a more natural and more efficient access to the same features as contained in the namespace "mvIMPACT_NET.acquire" of "mvIMPACT_NET.dll", which is why the latter one should only be used for backward compatibility but NOT when developing a new application.
- 6 Part of Micro-Manager.
## 1.2.2.2 NeuroCheck support

A couple of devices are supported by NeuroCheck. However between NeuroCheck 5.x and NeuroCheck 6.x there has been a breaking change in the internal interfaces. Therefore also the list of supported devices differs from one version to another and some additional libraries might be required.

For NeuroCheck 5.x the following devices are supported:

<table>
<thead>
<tr>
<th>Device</th>
<th>Additional software needed</th>
</tr>
</thead>
<tbody>
<tr>
<td>mvTITAN-G1</td>
<td>mvSDK driver for mvTITAN/mvGAMMA devices</td>
</tr>
<tr>
<td>mvTITAN-CL</td>
<td>mvSDK driver for mvTITAN/mvGAMMA devices</td>
</tr>
<tr>
<td>mvGAMMA-CL</td>
<td>mvSDK driver for mvTITAN/mvGAMMA devices</td>
</tr>
<tr>
<td>mvBlueFOX</td>
<td>mvIMPACT Acquire driver for mvBlueFOX devices, &quot;NCUSBmvBF.dll&quot;</td>
</tr>
</tbody>
</table>

For NeuroCheck 6.0 the following devices are supported:

<table>
<thead>
<tr>
<th>Device</th>
<th>Additional software needed</th>
</tr>
</thead>
<tbody>
<tr>
<td>mvTITAN-G1</td>
<td>mvIMPACT Acquire driver for mvTITAN/mvGAMMA devices</td>
</tr>
<tr>
<td>mvTITAN-CL</td>
<td>mvIMPACT Acquire driver for mvTITAN/mvGAMMA devices</td>
</tr>
<tr>
<td>mvGAMMA-CL</td>
<td>mvIMPACT Acquire driver for mvTITAN/mvGAMMA devices</td>
</tr>
<tr>
<td>mvHYPERION-CLb</td>
<td>mvIMPACT Acquire driver for mvHYPERION devices</td>
</tr>
<tr>
<td>Every other mvIMPACT Acquire compliant device</td>
<td>mvIMPACT Acquire driver for the corresponding device family, &quot;mv_impact.acquire.NeuroCheck6_*.dll&quot; (comes with the driver package, but the driver package must be installed AFTER installing NeuroCheck 6)</td>
</tr>
</tbody>
</table>

For NeuroCheck 6.1 the following devices are supported:

<table>
<thead>
<tr>
<th>Device</th>
<th>Additional software needed</th>
</tr>
</thead>
<tbody>
<tr>
<td>mvTITAN-G1</td>
<td>mvIMPACT Acquire driver for mvTITAN/mvGAMMA devices</td>
</tr>
<tr>
<td>mvTITAN-CL</td>
<td>mvIMPACT Acquire driver for mvTITAN/mvGAMMA devices</td>
</tr>
<tr>
<td>mvGAMMA-CL</td>
<td>mvIMPACT Acquire driver for mvTITAN/mvGAMMA devices</td>
</tr>
<tr>
<td>mvHYPERION-CLb</td>
<td>mvIMPACT Acquire driver for mvHYPERION devices</td>
</tr>
<tr>
<td>Every other mvIMPACT Acquire compliant device</td>
<td>mvIMPACT Acquire driver for the corresponding device family, &quot;mv_impact.acquire.NeuroCheck6_*--1.dll&quot; (comes with the driver package, but the driver package must be installed AFTER installing NeuroCheck 6.1)</td>
</tr>
</tbody>
</table>

## 1.2.2.3 VisionPro support

Every mvIMPACT Acquire driver package under Windows comes with an adapter to VisionPro from Cognex. The installation order does not matter. After the driver package and VisionPro has been installed, the next time VisionPro is started it will allow selecting the mvIMPACT Acquire device. No additional steps are needed.

MATRIX VISION devices that also comply with the GigE Vision or USB3 Vision standard don’t need any software at all, but can also use VisionPro’s built-in GigE Vision or USB3 Vision support.
1.2 How to get started?

1.2.2.4 HALCON support

HALCON comes with built-in support for mvIMPACT Acquire compliant devices, so once a device driver has been installed for the mvIMPACT Acquire device, it can also be operated from a HALCON environment using the corresponding acquisition interface. No additional steps are needed.

MATRIX VISION devices that also comply with the GigE Vision or USB3 Vision standard don’t need any software at all, but can also use HALCON’s built-in GigE Vision or USB3 Vision support. As some mvIMPACT Acquire device driver packages also come with a GenTL compliant interface, these can also be operated through HALCON’s built-in GenTL acquisition interface.

1.2.2.5 LabVIEW support

Every mvIMPACT Acquire compliant device can be operated under LabVIEW through an additional set of VIs which is shipped by MATRIX VISION as a separate installation ("mvLabVIEW Acquire"). MATRIX VISION devices that also comply with the GigE Vision or USB3 Vision standard don’t need any additional software at all, but can also be operated through LabVIEW’s GigE Vision or USB3 Vision driver packages.

1.2.2.6 DirectShow support

Every mvIMPACT Acquire compliant device driver package comes with an interface to DirectShow. In order to be usable from a DirectShow compliant application, devices must first be registered for DirectShow support. How to this is explained here (p.115).

1.2.2.7 Micro-Manager support

Every mvIMPACT Acquire compliant device can be operated under http://micro-manager.org when using mvIMPACT Acquire 2.18.0 or later and at least Micro-Manager 1.4.23 build AFTER 15.12.2016. The adapter needed is part of the Micro-Manager release. Additional information can be found here: http://micro-manager.org/wiki/MatrixVision.

1.2.3 Image acquisition concept

The image acquisition is based on queues to avoid the loss of single images. With this concept you can acquire images via single acquisition or triggered acquisition. For detailed description of the acquisition concept, please have a look at "How the capture process works" in the mvIMPACT_Acquire_API manual matching the programming language you are working with.

1.2.4 Installation

To install the frame grabber properly you have to follow these steps:
(Please follow the links for detailed descriptions.)

- **Windows:**
  - Please check the system requirements (p.15).
  - Please install the software and driver (p.16).
  - Please install the hardware (p.15).

- **Linux:**
  - Please check the system requirements (p.20).
  - Please install the software and driver (p.21).
  - Please install the hardware (p.15).
1.2.5 Programming

To control the camera and handle the images, you will have a good introduction by reading the main pages of the "mvIMPACT Acquire" interface references. Additionally, please have a look at the example programs. Several basic examples are available. The separate mvIMPACT Acquire manuals

- "mvIMPACT_Acquire_API_CPP_manual.chm",
- "mvIMPACT_Acquire_API_C_manual.chm", and
- "mvIMPACT_Acquire_API_NET_manual.chm"

are available as downloads from our website http://www.matrix-vision.com.
This document assumes a general knowledge of PCs and programming.

Since the documentation is published electronically, an updated version may be available online. For this reason we recommend checking for updates on the MATRIX VISION website.

MATRIX VISION cannot guarantee that the data is free of errors or is accurate and complete and, therefore, assumes no liability for loss or damage of any kind incurred directly or indirectly through the use of the information of this document.

MATRIX VISION reserves the right to change technical data and design and specifications of the described products at any time without notice.

Copyright

MATRIX VISION GmbH. All rights reserved. The text, images and graphical content are protected by copyright and other laws which protect intellectual property. It is not permitted to copy or modify them for trade use or transfer. They may not be used on websites.

- Windows® XP, Windows® Vista, Windows® 7 are trademarks of Microsoft, Corp.
- Linux® is a trademark of Linus Torvalds.

All other product and company names in this document may be the trademarks and tradenames of their respective owners and are hereby acknowledged.

The manual has been generated with Doxygen (Website: http://www.doxygen.org).

Parts of the log file creation and the log file display make use of Sarissa (Website: http://dev.abiss.gr/sarissa) which is distributed under the GNU GPL version 2 or higher, GNU LGPL version 2.1 or higher and Apache Software License 2.0 or higher. The Apache Software License 2.0 is part of this driver package.
## 3 Revisions

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>21. December 2016</td>
<td>Added Setting up multiple display support and/or work with several capture settings in parallel (p.61).</td>
</tr>
<tr>
<td>21. October 2014</td>
<td>Added description about the record mode in <em>How to see the first image</em> (p.50).</td>
</tr>
<tr>
<td>06. December 2013</td>
<td>Added information about Changing the view of the property grid to assist writing code that shall locate driver features (p.69).</td>
</tr>
<tr>
<td>15. October 2013</td>
<td>Added <em>Webcasts</em> (p.10) links.</td>
</tr>
<tr>
<td></td>
<td>Added chapter <em>Bit-shifting an image</em> (p.88).</td>
</tr>
<tr>
<td>20. September 2012</td>
<td>Added chapter &quot;Porting existing code written with versions earlier than 3.0.0&quot;</td>
</tr>
<tr>
<td>17. July 2012</td>
<td><em>Firmware Update</em> (p.100): Corrected &quot;Switch 3&quot; to &quot;Switch 1&quot;.</td>
</tr>
<tr>
<td></td>
<td>Added use case <em>Working with a Basler Sprint line scan color camera</em> (p.126).</td>
</tr>
<tr>
<td></td>
<td>Added use case <em>Synchronous acquisition with different camera settings</em> (p.139).</td>
</tr>
<tr>
<td>16. February 2012</td>
<td>Renewed chapter <em>wxPropView</em> (p.48).</td>
</tr>
<tr>
<td>09. November 2011</td>
<td>Added <em>Settings behavior during startup</em> (p.25) in chapter <em>Quickstart</em> (p.15).</td>
</tr>
<tr>
<td>26. July 2011</td>
<td>Removed chapter <em>EventHandler</em>. See &quot;Porting existing code written with versions earlier than 2.0.0&quot;.</td>
</tr>
<tr>
<td>11. July 2011</td>
<td>Added chapter &quot;Callback demo&quot;.</td>
</tr>
<tr>
<td>06. June. 2011</td>
<td>Added chapter &quot;Porting existing code written with versions earlier than 2.0.0&quot;.</td>
</tr>
<tr>
<td>28. March 2011</td>
<td>Added LED description for mvHYPERION-CLx frame grabbers (p.28).</td>
</tr>
<tr>
<td>18. January 2011</td>
<td>Added chapter <em>Setting up multiple display support and/or work with several capture settings in parallel</em> (p.61).</td>
</tr>
<tr>
<td>07. December 2010</td>
<td>Added chapter <em>How to allocate image memory</em> (p.100).</td>
</tr>
<tr>
<td>19. October 2010</td>
<td>Added chapter &quot;Chunk data format&quot;.</td>
</tr>
<tr>
<td>01. Oct. 2010</td>
<td>Updated <em>Working with trigger events</em> (p.134) and Camera acquisition techniques (p.89).</td>
</tr>
<tr>
<td></td>
<td>Added chapter <em>Working with an rotary encoder</em> (p.123).</td>
</tr>
<tr>
<td>01. Oct. 2010</td>
<td>Updated <em>Components</em> (p.44) table of (mvHYPERION-HD-SDI-2) and added supported signal formats.</td>
</tr>
<tr>
<td>17. Sep. 2010</td>
<td>Corrected image of connector J6 <em>Connectors</em> (p.28) (mvHYPERION-CLx) and <em>Connectors</em> (p.42) (mvHYPERION-HD-SDI).</td>
</tr>
<tr>
<td>19. Apr. 2010</td>
<td>Added example <em>ContinuousCaptureDirectX</em>.</td>
</tr>
<tr>
<td>13. Jan. 2010</td>
<td>Added chapter &quot;Porting existing code written with versions earlier then 1.12.0&quot;.</td>
</tr>
<tr>
<td>10. Nov. 2009</td>
<td>Added Windows 7 as supported operating system.</td>
</tr>
<tr>
<td>Date</td>
<td>Revision</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>18. Aug. 2008</td>
<td>Added new example CaptureToUserMemory.</td>
</tr>
<tr>
<td>11. July 2008</td>
<td>Corrected Figure of J6 connector Technical data (p.28).</td>
</tr>
<tr>
<td>29. Apr. 2008</td>
<td>Added chapter How to recover a broken firmware update (p.100).</td>
</tr>
<tr>
<td>12. Dec. 2007</td>
<td>Added chapter &quot;Pinning J6 (internal digital I/Os)&quot; in Connectors (p.28).</td>
</tr>
<tr>
<td>06. Dec. 2007</td>
<td>Updated chapter What's inside and accessories (p.14).</td>
</tr>
<tr>
<td>07. Nov. 2007</td>
<td>Added chapter &quot;Porting existing code written with versions earlier then 1.10.0&quot;.</td>
</tr>
<tr>
<td>25. Sep. 2007</td>
<td>Added chapter mvDeviceConfigure (p.95) and description about &quot;Switches&quot; in Connectors (p.28).</td>
</tr>
<tr>
<td>1. August 2007</td>
<td>Rewritten &quot;How to use this manual&quot;. This book now includes a getting started chapter (see: Composition of the manual (p.2)).</td>
</tr>
<tr>
<td>Feb. 2007</td>
<td>Initial version</td>
</tr>
</tbody>
</table>
4 Graphic Symbols

4.1 Notes, Warnings, Attentions

Note

A note indicates important information that helps you optimize usage of the products.

Warning

A warning indicates how to avoid either potential damage to hardware or loss of data.

Attention

An attention indicates a potential for property damage, personal injury, or death.

All due care and attention has been taken in preparing this manual. In view of our policy of continuous product improvement, however, we can accept no liability for completeness and correctness of the information contained in this manual. We make every effort to provide you with a flawless product.

In the context of the applicable statutory regulations, we shall accept no liability for direct damage, indirect damage or third-party damage resulting from the acquisition or operation of a MATRIX VISION product. Our liability for intent and gross negligence is unaffected. In any case, the extend of our liability shall be limited to the purchase price.

4.2 Webcasts

This icon indicates a webcast about an issue which is available on our website.
5 Important information

We cannot and do not take any responsibility for the damage caused to you or to any other equipment connected to the mvHYPERION frame grabber. Similarly, warranty will be void, if a damage is caused by not following the manual.

Handle the mvHYPERION frame grabber with care. Do not misuse the mvHYPERION frame grabber. Avoid shaking, striking, etc. The mvHYPERION frame grabber could be damaged by faulty handling or shortage.

• Handle with care and avoid damage of electrical components by electrostatic discharge (ESD):
  - Discharge body static (contact a grounded surface and maintain contact).
  - Avoid all plastic, vinyl, and styrofoam (except antistatic versions) around printed circuit boards.
  - Do not touch components on the printed circuit board with your hands or with conductive devices.

5.1 European Union Declaration of Conformity statement

The mvHYPERION-CLx is in conformity with all applicable essential requirements necessary for CE marking. It corresponds to the EU EMC guideline 2004/108/EC based on the following harmonized standards Electromagnetic compatibility (EMC)


• Interference immunity EN 55022 : 2006 + A1:2007 Class A

• Interference immunity EN 55022 : 2006 + A1:2007 Class B with modifications

EN 55022 : 2006 + A1:2007 Class B with modifications requires an CameraLink cable with an retrofittable ferrite to be used (near to frame grabber connector) such as

  - Company: Würth Elektronik Type: WE No. 742 711 31

MATRIX VISION corresponds to the EU guideline WEEE 2002/96/EG on waste electrical and electronic equipment and is registered under WEEE-Reg.-No. DE 25244305.
EG-Konformitätserklärung
EC Declaration of conformity

Der Hersteller
The Manufacturer
Matrix Vision GmbH
Talstraße 16
71570 Oppenweiler
Germany

erklärt hiermit, dass sein Produkt:
hereby declares, that his product:

Geräteart:
Device:
Digitaler Frame Grabber
Digital frame grabber

Typbezeichnung:
Type:
mvHYPERION-CLb
mvHYPERION-CLe

mit den Bestimmungen folgender Europäischer Richtlinien übereinstimmt:
complies with the provisions of the following European Directives:

auf Grundlage folgender harmonisierter Normen:
based on the following harmonized standards:

Elektromagnetische Verträglichkeit (EMV) / Electromagnetic compatibility (EMC)
Störsendung / Interference emission
EN 55022 : 2006 + A1 : 2007 (Klasse A / Class A)

Störfestigkeit / Interference immunity

Oppenweiler, 22.06.2009

I.V. Uwe Hagmatz
Leiter Entwicklung / Head of Development

Bitte beachten: Um die Einhaltung obiger Normen sicherzustellen, ist die Verwendung
high-quality, shielded connecting cables is required.
von hochwertiger; vollständig geschirmter Anschlusskabel unbedingt erforderlich.

Please note: in order to fulfill the above standards, the use of high-quality, shielded connecting
Konformitätserklärung
Declaration of conformity

Der Hersteller  Matrix Vision GmbH
The Manufacturer  Tulstrasse 16
                              71570 Oppenweiler
                              Germany

erklärt hiermit, dass sein Produkt
herewith declares, that its product

Typbezeichnung:  mvHYPERION-CLf/-CLm
Type:  mvHYPERION-CLf/-CLm

mit den Bestimmungen folgender Europäischer Richtlinien übereinstimmt:
complies with the provisions of the following European Directives:

auf Grundlage folgender harmonisierter Normen:
based on the following harmonised standards:

Elektromagnetische Verträglichkeit (EMV) / Electromagnetic compatibility (EMC)
Störmessung / Interference emission

Störfestigkeit / Interference immunity

Oppenweiler, 17.06.2011

I.V. Uwe Hagemeier
Leiter Entwicklung / Head of Development

Bitte beachten. Um die Einhaltung obiger Normen sicherzustellen, ist die Verwendung
high-voltage, well-shielded and properly grounded connecting cables is required.
6 Introduction

The mvHYPERION-Series are frame grabbers for the bus system PCI Express®. The mvHYPERION frame grabber series for PCI Express® offers image processing with fast cameras using maximum capture bandwidth up to 1 Gbps. Depending on the model type, the frame grabbers are suitable for high-end machine vision applications with CameraLink cameras as well as broadcasting or surveillance solutions.

![Figure 1: mvHYPERION series](image)

There are digital inputs available for external synchronization and digital outputs for e.g. controlling a flash.

The mvHYPERION series is suitable for following application areas:

<table>
<thead>
<tr>
<th>Area/Application</th>
<th>32R16</th>
<th>HD-SDI</th>
<th>CLx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Industrial Image Processing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2D/3D measurement</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OCR, Pattern recognition</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>control, robotics</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>visualisation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>line camera applications</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Security</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>object surveillance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>traffic surveillance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>compressed recording</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>video sensoric</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microscopy/Diagnostic</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>light microscopy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>laser scan systems</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>electron microscopy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Medicine</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>visual diagnostic</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>laboratory systems</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image Capture/Recording</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>document line scanner</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>high resolution cameras</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>broadcasting</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Figure 2: Application areas](image)

6.1 What's inside and accessories

Due to the varying fields of application the mvHYPERION series is shipped without accessories. The package contents:

- mvHYPERION frame grabber

Accessories for the mvHYPERION-CLx frame grabbers:
7 Quickstart

7.1 Hardware installation

**Warning**

Please take all proper Electro Static Discharge (ESD) precautions during the installation of your new hardware!

Before starting the installation, turn off your computer and all peripheral devices. Disconnect the computer from the power supply and all necessary components.

**Note**

To avoid doing damage to the hardware, discharge yourself of static charge by touching e.g. the casing. Beware of touching contacts of the frame grabber or of the computer.

- Select a free busmaster slot (PCI Express). Remove the slot's cover at the back of the computer and keep the screw.
- Carefully insert the board into the slot by holding the board at the top and gently pushing both ends into the slot at the same time. Press onto the upper edge of the board to make sure it is seated in the slot firmly.
- Do not force the board into the slot! You run the risk of bending the contacts. If the board does not fit easily, pull it back out, and try again.
- Fasten the board's bracket at the back of the computer using the screws you saved from the shield.
- Put the cover back on the computer and reconnect the peripheral devices.
- Start the computer.

**Warning**

According to the construction, if you want to connect or disconnect a PoCL (p. 119) camera, please be sure that the PC or the mvHYPERION frame grabber is switched off! Otherwise, during the connection, the camera or the frame grabber could be short-circuited and possibly destroyed!

7.2 Windows

7.2.1 System Requirements

Currently supported Windows versions are:

- Microsoft Windows 7 (32-bit, 64-bit) (requires min. 2 GB main memory)
- Microsoft Windows 8.1 (32-bit, 64-bit) (requires min. 2 GB main memory)

Consecutively the installation for Windows will be described. The description for the Linux installation can be found here: Linux (p. 20).

**Note**

For a correct installation of the frame grabber please install the MSI package before connecting any board to the system. Afterwards you can install the physical board(s) and when the system starts again everything else is done automatically.
7.2.2 Software installation

All necessary drivers for Windows and Linux are contained in the mvIMPACT CD-ROM or DVD-ROM. For newer driver versions we recommend to visit the MATRIX VISION website at www.matrix-vision.de, section Support/Download/Hardware.

After the Hardware installation (p. 15) the boot sequence shows "Found New Hardware" and starts the Windows Hardware Wizard. Close this window and insert the mvIMPACT CD-ROM or DVD-ROM into your drive and select "Driver installation ..." and the needed mvIMPACT Acquire driver (e.g. "mvTITAN / mvGAMMA").

Figure 1: Start window

After the click on the needed driver the installation process starts.
Select the folder, where you want to install the software.

Select the features, which you want to install. Following features exist:
• "Base Libraries"
  This feature contains all necessary files for property handling and display. Therefore, it is not selectable.

• "mvHYPERION driver"
  This is also not selectable.

• "Tools"
  This feature contains tools for the frame grabber (e.g. to acquire images (wxPropView (p. 48)).

• "mvIMPACT acquire API"
  The "mvIMPACT acquire API" contains the header for own programming. Additionally you can choose the examples, which installs the sources of wxPropView, mvIPConfigure and various small examples. The project files shipped with the examples have been generated with Visual Studio 2013. However projects and make-files for other compilers can be generated using CMake fairly easy. See CMake section in the C++ manual for additional details.

• "Documentation"
  This will install this manual as single HTML help file (.chm).

Figure 4: mvHYPERION installer - Select features

Confirm the installation by clicking "Next".
The installation process copies the files to Windows. Then Windows shows a message to signal that this driver is not checked through Microsoft. This is only an attempt to make insecure and it is recommended to ignore it.

Press "Continue Anyway" and finish the driver installation.
After this, you have to restart the system. Afterwards, you can acquire images with the frame grabber. Simply start the application wxPropView (p.48) (wxPropView.exe).

See also

wxPropView (p.48)

7.3 Linux

7.3.1 System Requirements

Kernel requirements

Kernel 2.6.x.

Software requirements
• Linux x86 (32-bit)
  – The 32 bit version will run on a 64-bit Linux system if the other library requirements are met with 32-bit libraries. i.e. you cannot mix 64 and 32-bit libraries and applications.
  – Versions for Linux on x86-64 (64-bit), PowerPC, ARM or MIPS may be possible on request.
• GNU compiler version GCC 3.2.x or greater and associated toolchain.

Other requirements

• libexpat (http://expat.sourceforge.net)
• Optional: wxWidgets 2.6.x (non Unicode) for the wxWidget test programs.
• Optional: udev or hotplug subsystem (see also 6. below).

As an example of which packets need to be installed, consider OpenSuSE 10.1:

• The compiler used is gcc 4.1.0 and may need to be installed. Use the "gcc" und "gcc-c++" RPMs. Other RPMs may be installed automatically due to dependencies (e.g. make).
• libexpat will almost definitely be installed already in any software configuration. The RPM is called "expat".
• Install the wxWidgets "wxGTK" and "wxGTK-develop" RPMs. Others that will be automatically installed due to dependencies include "wxGTK-compat" and "wxGTK-gl". Although the MATRIX VISION software does not use the ODBC database API the SuSE version of wxWidgets has been compiled with ODBC support and the RPM does not contain a dependency to automatically install ODBC. For this reason you must also install the "unixODBC-devel" RPM.
• OpenSuSE 10.1 uses the udev system so a separate hotplug installation is not needed.

Hardware requirements

PC with PCI Express Single lane

Note

The driver contains libraries for Linux x86 (32 bit) or Linux 64-bit (x86_64). There are separate package files for systems with toolchains based on GNU gcc 3.2.x - 3.3.x and those based on GNU gcc >= 3.4.x. gcc 3.1.x may work but, in general, the older your toolchain is, the less likely it is that it will work. Toolchains based on GNU gcc 2.x.x are not supported at all.

7.3.2 Installing the mvIMPACT Acquire driver

7.3.2.1 Using the installer script

To use the mvHYPERION frame grabber within Linux (grab images from it and change its settings), a driver is needed, consisting of several libraries and several configuration files. These files are required during run time.

To develop applications that can use the mvHYPERION frame grabber, a source tree is needed, containing header files, makefiles, samples, and a few libraries. These files are required at compile time.

Both file collections are distributed in a single package:

mvHYPERION-x86_ABI2-n.n.n.tgz
1. Please start a console and change into a directory e.g. /home/username/workspace

   cd /home/username/workspace

2. Copy the install script and the hardware driver to the workspace directory (e.g. from a driver CD or from the website):

   ~/workspace$ cp /media/cdrom/drv/Linux/install_mvHYPERION.sh / .
   && cp /media/cdrom/drv/Linux/mvHYPERION-x86_ABI2-2.14.0.tgz -t ./

3. Run the install script:

   ~/workspace$ ./install_mvHYPERION.sh

   **Note**

   The install script has to be executable. So please check the rights of the file.
   During installation, the script will ask, if it should build all tools and samples.
   You may need to enable the execute flag with

   chmod a+x install_mvHYPERION.sh

The installation script checks the different packages and installs them with the respective standard packages manager (apt-get) if necessary.

   **Note**

   The installation script ("install_mvHYPERION.sh") and the archive ("mvHYPERION-x86_ABI2-2.14.0.tgz") must reside in the same directory. Nothing is written to this directory during script execution, so no write access to the directory is needed in order to execute the script.

You need Internet access in case one or more of the packages on which the GenICam™ libs depend are not yet installed on your system. In this case, the script will install these packages, and for that, Internet access is required.

The script takes two arguments, both of which are optional:

1. target directory name
2. version

   **Note**

   This directory is only used for the files that are run time required.

The files required at compile time are always installed in "$HOME/mvimpact-acquire-n.n.n". The script also creates a convenient softlink to this directory:

   mvimpact-acquire -> mvIMPACT_acquire-2.3.2

If this argument is not specified, or is ". ", the driver will be placed in the current working directory.

The version argument is entirely optional. If no version is specified, the most recent mvHYPERION-x86_ABI2-2.14.0.tgz found in the current directory will be installed.

If you have also installed tools and samples, you will find them in apps. Just change into the subdirectory, for example, "apps/mvPropView/x86" you can find the camera setup tool wxPropView (p. 33). You can find it in subdirectory "~/mvimpact-acquire/apps/mvPropView/x86". However, you do not always have to start the tool from this folder. The installer script has created symbolic links so that it is enough to type in wxPropView throughout the system to start wxPropView (p. 33).

The mvIMPACT Acquire libraries look for camera definition files in the directory "mvimpact-acquire/camerafiles" so you will need to create these directories as the "root" user using "mkdir -p ./mvimpact-acquire/camerafiles".
7.3.2.2 Doing it manually

The mvHYPERION is controlled by a number of user-space libraries. It is not necessary to compile a kernel module.

1. Logon to the PC as the "root" user or start a super user session with "su". Start a console with "root" privileges.

2. Determine which package you need by issuing the following command in a terminal window:

   gcc -v

   This will display a lot of information about the GNU gcc compiler being used on your system. In case of the version number you have to do following:

3. You can now install the mvHYPERION libraries as follows:

   • create a new directory somewhere on your system.
   • copy the correct package file to this directory and change into this directory with "cd".

   The libraries are supplied as a "tgz" archive with the extension ".tgz".

   (a) Unpack the archive using "tar" e.g.:

   tar xvzf mvhyperion-x86-ABI1-1.8.4.55.tgz

   or

   tar xvzf mvhyperion-x86-ABI2-1.8.4.55.tgz

   (b) After installing the access libraries you will see something like the following directory structure in your directory (dates and file sizes will differ from the list below):

   drwxr-xr-x 10 root root 4096 Jan 5 15:08 .
   drwxr-xr-x 23 root root 4096 Jan 4 16:33 ..
   drwxr-xr-x 3 root root 4096 Jan 5 15:08 DriverBase
   -rw-r--r-- 1 root root 1079 Jan 5 15:08 Makefile
   drwxr-xr-x 7 root root 4096 Jan 5 15:08 apps
   drwxr-xr-x 4 root root 4096 Jan 5 15:08 common
   drwxr-xr-x 3 root root 4096 Jan 5 15:08 lib
   drwxr-xr-x 3 root root 4096 Jan 5 15:08 mvDeviceManager
   drwxr-xr-x 2 root root 4096 Jan 5 15:08 mvIMPACT_CPP
   drwxr-xr-x 3 root root 4096 Jan 5 15:08 mvPropHandling
   drwxr-xr-x 1 root root 4096 Jan 5 15:08 scripts

   The directory "lib/x86" contains the pre-compiled 32-bit libraries for accessing the mvBlueFOX. If 64-bit libraries are supplied they will be found in "lib/x86_64". The "apps" directory contains test applications (source code). The directories contain headers needed to write applications for the device.

   Since the libraries are not installed to a directory known to the system i.e. not in the "ldconfig" cache you will need to tell the system where to find them by...

   • using the "LD_LIBRARY_PATH" environment variable,
   • or copying the libraries by hand to a system directory like "/usr/lib" (or using some symbolic links),
   • or entering the directory in "/etc/ld.so.conf" and running "ldconfig".

   e.g. to start the application called "LiveSnap":

   MATRIX VISION GmbH
Please declare the device e.g. HC+ or HC000000001

cd my_directory
LD_LIBRARY_PATH='pwd'/lib/x86 apps/LiveSnap/x86/LiveSnap HC+

For 64-bit it will look like this...
LD_LIBRARY_PATH='pwd'/lib/x86_64 apps/LiveSnap/x86_64/LiveSnap HC+

For ARM it will look like this...
LD_LIBRARY_PATH='pwd'/lib/arm apps/LiveSnap/arm/LiveSnap HC+

etc.

After installing the libraries and headers you may continue with "3." below as a normal user i.e. you do not need to be "root " in order to compile the test applications. See also the note "4." below.

To build the test applications type "make". This will attempt to build all the test applications contained in "apps". If you have problems compiling the wxWidget library or application you may need to do one or more of the following:

• install the wxWidget 3.x development files (headers etc.) supplied for your distribution. (See "Other requirements" above).

• fetch, compile and install the wxWidget 3.x packet from source downloaded from the website (http://www.wxwidgets.org).

• alter the Makefiles so as to find the wxWidget configuration script called wx-config.

The files you may need to alter are to be found here:

apps/mvPropView/Makefile.inc

You will find the compiled test programs in the subdirectories "apps/.../x86". For 64 bit systems it will be "apps/.../x86_64". For ARM systems it will be "apps/.../arm".

If you cannot build the wxWidget test program you should, at least, be able to compile the text-based test programs in apps/SingleCapture, apps/LiveSnap, etc.

The Makefile will also attempt to configure itself so that the mvHYPERION kernel module can be built. You should see the following message at the end of the compile block:

===============================================================================
To install the mvHYPERION kernel module now make sure that you are root and type: make install
===============================================================================

If you are not already logged in as the "root " user you must now use "su " to change users and type "make install". On an Ubuntu system you might try "sudo make install". The kernel module will now be built and installed.

Now you will have to tell your system to use this kernel module and to associate it with a device node. To do this you need to edit the file "/etc/modprobe.conf". Depending on your system you may have a directory called "/etc/modules.d/", where you can put files that are included automatically in "/etc/modprobe.conf". Other systems (e.g. older SuSE) use a file called "/etc/modprobe.conf.local" which the user may alter. Which ever way you do it, you need to add some lines like this:

alias char-major-64 hyperion
options hyperion major_dev_num=64

Afterwards, please use "depmod -a" to tell the system about this change. The lines above tell the system to use a device called "/dev/hyperion" with a major number of 64.

(d) If you are using a system with an up-to-date version of udev you might be interested in the file "Scripts/50-udev-hyperion.rules". By including this file in your udev rules directory you can tell your system to create a device node for the mvHYPERION automatically when loading the kernel module. You will need to edit the file to fit your system. As delivered, all entries are commented out.

If you do not use udev then you will have to create a device node yourself by hand. For example, you could do the following to use major device number 64:

mknod /dev/hyperion c 64 0
If you have more than one mvHYPERION in your PC you will need a device node per card:

```bash
mknod /dev/hyperion0 c 64 0
mknod /dev/hyperion1 c 64 1
mknod /dev/hyperion2 c 64 2
mknod /dev/hyperion3 c 64 3
```

(e) The mvIMPACT Acquire libraries look for camera definition files in the directory "/etc/matrix-vision/mvimpact-acquire/camerafiles" so you will need to create these directories as the "root" user using "mkdir -p /etc/matrix-vision/mvimpact-acquire/camerafiles". You should download the camera definitions for your camera from the MATRIX VISION website and copy them to this directory.

### 7.4 Connecting a camera

To connect a camera, for example via CameraLink cable to the mvHYPERION-CLx frame grabber, please do the following:

- Connect the cable from the camera (labeled with e.g. Data channel 1) to the first connector J1 of the mvHYPERION frame grabber.
- Optionally, if you are using a Medium or Full, connect the cable from the camera (labeled with e.g. Data channel 2) to the second connector J2 of the mvHYPERION.
- Optionally, use the connector J3 to power the camera.

Make sure that you do not mix up the channels. For this, please have a look at chapter [Technical data](#) where to find the specific connectors.

- Afterwards, start [wxPropView](#) and choose the "Generic" camera definition.
- Now, press the [Live](#) button - at this point you should see something from the camera.
- Then, create a new camera definition as described in [Working with camera descriptions](#).
- Finally, export the new camera definition and choose it in "wxPropView -> ImageSetting -> Camera -> Type".

### 7.5 Settings behavior during startup

Settings contain all the parameters that are needed to prepare and program the device for the image capture. Every image can be captured with completely different set of parameters. In almost every case, these parameters are accessible via a property offered by the device driver. A setting e.g. might contain

- the gain to be applied to the analog to digital conversion process for analog video sources or
- the AOI to be captured from the incoming image data.

So for the user a setting is the one and only place where all the necessary modifications can be applied to achieve the desired form of data acquisition.

Now, whenever a device is opened, the driver will execute following procedure:
• Please note that each setting location step in the figure from above internally contains two search steps. First the framework will try to locate a setting with user scope and if this can’t be located, the same setting will be searched with global (system-wide) scope. Under Windows® this e.g. will access either the HKEY_CURRENT_USER or (in the second step) the HKEY_LOCAL_MACHINE branch in the Registry.

• Whenever storing a product specific setting, the device specific setting of the device used for storing will be deleted (if existing). So when the user is currently working with a device ‘VD000001’ belonging to the product group 'VirtualDevice' and there is a setting exclusively for this device storing a product specific setting now will automatically delete the setting for ‘VD000001’. Otherwise a product specific setting would never be loaded as a device specific setting will always be found first.

• The very same thing will also happen when opening a device from any other application! wxPropView (p. 48) does not behave in a special way but only acts as an arbitrary user application.

• Whenever storing a device family specific setting, the device specific or product specific setting of the device used for storing will be deleted (if existing). See above to find out why.

• Under Windows® the driver will not look for a matching XML file during start-up automatically as the native storage location for settings is the Windows® Registry. This must be loaded explicitly by the user by using the appropriate API function offered by the SDK. However, under Linux XML files are the only setting formats understood by the driver framework thus here the driver will also look for them at start-up. The device specific setting will be an XML file with the serial number of the device as the file name, the product specific setting will be an XML file with the product string as the filename, the device family specific setting will be an XML file with the device family name as the file name. All other XML files containing settings will be ignored!

• Only the data contained in the lists displayed as "Image Setting", "Digital I/O" and "Device Specific Data" under wxPropView (p. 48) will be stored in these settings!

• Restoring of settings previously stored works in a similar way. After a device has been opened the settings will be loaded automatically as described above.

• A detailed description of the individual properties offered by a device will not be provided here but can be found in the C++ API reference, where descriptions for all properties relevant for the user (grouped together in classes sorted by topic) can be found. As wxPropView (p. 48) doesn’t introduce new functionality but simply evaluates the list of features offered by the device driver and lists them any modification made using the GUI
controls just calls the underlying function needed to write to the selected component. \textit{wxPropView} (p. 48) also doesn’t know about the type of component or e.g. the list of allowed values for a property. This again is information delivered by the driver and therefore can be queried by the user as well without the need to have special\textit{ inside information}. One version of the tool will always be delivered in source so it can be used as a reference to find out how to get the desired information from the device driver.
8 Technical data

8.1 mvHYPERION-CLx

8.1.1 Block diagrams

The following block diagrams show schematically how the different mvHYPERION-CLx are designed.

Figure 1: mvHYPERION-CLb block diagram | mvHYPERION-CLe block diagram

Figure 2: mvHYPERION-CLm block diagram | mvHYPERION-CLf block diagram

8.1.2 Connectors

The mvHYPERION supports the serial communication over CameraLink™ cable as described in the CameraLink™ specification. The driver offers a serial interface without the need of a host PC’s COM port. During the boot sequence of the operating system the serial interface is initialized. Normally, manufacturers of CameraLink™ cameras provide software to parameterize the camera. If this software abides by the specification, it will access our serial interface driver automatically.
For Linux there is no CameraLink™ specified library. Therefore we ship CameraLink™ compliant library libclserMV.so which can be found in the lib directory.

8.1.2.1 Status LEDs

<table>
<thead>
<tr>
<th>LED</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D9</td>
<td>FPGA state</td>
<td>Green: FPGA is loaded</td>
</tr>
<tr>
<td>D10</td>
<td>PCI Express® connection state</td>
<td>Green: No problem with connection</td>
</tr>
</tbody>
</table>

Figure 3: mvHYPERION-CLe

Figure 4: mvHYPERION-CLf

Figure 5: Rev. 1.x

Figure 6: Rev. 3.x
<table>
<thead>
<tr>
<th>LED</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D9</td>
<td>FPGA state</td>
<td>Green: FPGA is loaded</td>
</tr>
<tr>
<td>D10</td>
<td>PCI Express® x4 connection state</td>
<td>Green: No problem with connection (PCI Express host supports 4 lanes)</td>
</tr>
<tr>
<td>D24</td>
<td>PCI Express® x1 connection state</td>
<td>Green: No problem with connection (PCI Express host supports 1 lane)</td>
</tr>
</tbody>
</table>

8.1.2.2 Use of J1..J4

<table>
<thead>
<tr>
<th>Connector usage</th>
<th>mvHYPERION</th>
<th>-CLb</th>
<th>-CLE</th>
<th>-CLm</th>
<th>-CLf</th>
</tr>
</thead>
<tbody>
<tr>
<td>J1</td>
<td>Camera 1 (BASE 1)</td>
<td>Camera 1 (BASE 1)</td>
<td>Camera 1 (BASE 1)</td>
<td>Camera 1 (BASE 1)</td>
<td></td>
</tr>
<tr>
<td>J2</td>
<td>Camera 1 trigger / sync / strobe / power connector</td>
<td>Camera 1 trigger / sync / strobe / power connector</td>
<td>Camera 1 trigger / sync / strobe / power connector</td>
<td>Camera 1 trigger / sync / strobe / power connector</td>
<td></td>
</tr>
<tr>
<td>J3</td>
<td>Camera 1 trigger / sync / strobe / power connector</td>
<td>Camera 1 trigger / sync / strobe / power connector</td>
<td>Camera 1 trigger / sync / strobe / power connector</td>
<td>Camera 1 trigger / sync / strobe / power connector</td>
<td></td>
</tr>
<tr>
<td>J4</td>
<td>Camera 2 trigger / sync / strobe / power connector</td>
<td>Camera 2 trigger / sync / strobe / power connector</td>
<td>Camera 2 trigger / sync / strobe / power connector</td>
<td>Camera 1 trigger / sync / strobe / power connector</td>
<td></td>
</tr>
</tbody>
</table>

8.1.2.3 Pinning J1/J2 (CL configuration)

Figure 7: Mini CameraLink connector (female)

<table>
<thead>
<tr>
<th>Pin J1/J2</th>
<th>(used as) BASE</th>
<th>(used as) MEDIUM</th>
<th>(used as) FULL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal</td>
<td>Type</td>
<td>Signal</td>
<td>Type</td>
</tr>
<tr>
<td>1</td>
<td>Internal shield or Power</td>
<td>PoCL (p.119) (+12V or Ground)</td>
<td>Internal shield or Power</td>
</tr>
<tr>
<td>14</td>
<td>Internal shield</td>
<td>Ground</td>
<td>Internal shield</td>
</tr>
<tr>
<td>25</td>
<td>X0-</td>
<td>Input 1-</td>
<td>Y0-</td>
</tr>
<tr>
<td>12</td>
<td>X0+</td>
<td>Input 1+</td>
<td>Y0+</td>
</tr>
<tr>
<td>24</td>
<td>X1-</td>
<td>Input 2-</td>
<td>Y1-</td>
</tr>
<tr>
<td>11</td>
<td>X1+</td>
<td>Input 2+</td>
<td>Y1+</td>
</tr>
<tr>
<td>23</td>
<td>X2-</td>
<td>Input 3-</td>
<td>Y2-</td>
</tr>
<tr>
<td>10</td>
<td>X2+</td>
<td>Input 3+</td>
<td>Y2+</td>
</tr>
<tr>
<td>22</td>
<td>Xclk-</td>
<td>Input 4-</td>
<td>Yclk-</td>
</tr>
<tr>
<td>21</td>
<td>X3-</td>
<td>Input 5-</td>
<td>Y3-</td>
</tr>
<tr>
<td>8</td>
<td>X3+</td>
<td>Input 5+</td>
<td>Y3+</td>
</tr>
<tr>
<td>20</td>
<td>SerTC+</td>
<td>Output 6+</td>
<td>Not used</td>
</tr>
</tbody>
</table>
8.1.2.4 Pinning J3 (Camera 1: Trigger/Flash/Power)

Figure 8: 8-pin Binder Line 711 (female)

<table>
<thead>
<tr>
<th>Pin.</th>
<th>Signal</th>
<th>Cable (KS99-0285)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+12 V DC (0.7A/2A) (camera power)</td>
<td>Red</td>
</tr>
<tr>
<td>2</td>
<td>Trigger-Out1 (+) -&gt; Collector</td>
<td>White</td>
</tr>
<tr>
<td>3</td>
<td>Trigger-Out1 (-) -&gt; Emitter</td>
<td>Brown</td>
</tr>
<tr>
<td>4</td>
<td>Trigger-In1 (+) -&gt; Anode</td>
<td>Green</td>
</tr>
<tr>
<td>5</td>
<td>Trigger-In1 (-) -&gt; Cathode</td>
<td>Yellow</td>
</tr>
<tr>
<td>6</td>
<td>Sync-In1 (+) -&gt; Anode</td>
<td>Gray</td>
</tr>
<tr>
<td>7</td>
<td>Sync-In1 (-) -&gt; Cathode</td>
<td>Pink</td>
</tr>
<tr>
<td>8</td>
<td>GND (camera power)</td>
<td>Blue</td>
</tr>
</tbody>
</table>

8.1.2.5 Pinning J4 (Camera 2: Trigger/Flash/Power)

<table>
<thead>
<tr>
<th>Pin.</th>
<th>Signal</th>
<th>Cable (KS99-0285)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+12 V DC (0.7A/2A) (camera power)</td>
<td>Red</td>
</tr>
</tbody>
</table>
Recommended plugs for 8-pin Binder series 711:

- 711: Binder ordering no. 99-0479-100-08 / 99-0479-102-08

Detailed information:  http://www.binder-connector.de

8.1.2.6 Pinning J5 (Power supply (floppy))

You can connect a free power supply cable for floppy drives on connector J5 to increase the available current on the power supply pins on J3 and J4 to 2A.

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Signal direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+12 V</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Ground</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Ground</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Not connected</td>
<td></td>
</tr>
</tbody>
</table>

8.1.2.7 Pinning J6 (internal digital I/Os)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Signal direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>1..5</td>
<td>used internally (do not connect!)</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Ground</td>
<td>Ground</td>
</tr>
<tr>
<td>7</td>
<td>+5V out</td>
<td>+5V DC power supply</td>
</tr>
<tr>
<td>8</td>
<td>+3.3V out</td>
<td>+3.3V DC power supply</td>
</tr>
<tr>
<td>9..12</td>
<td>GPIN0..3</td>
<td>LVTTL(3.3V) input. not 5V tolerant!</td>
</tr>
<tr>
<td>13..16</td>
<td>GPOUT0..3</td>
<td>LVTTL(3.3V) output. not 5V tolerant!</td>
</tr>
<tr>
<td>17</td>
<td>Ground</td>
<td>Ground</td>
</tr>
<tr>
<td>18</td>
<td>+12V DC Out</td>
<td>+12V DC power supply</td>
</tr>
<tr>
<td>19</td>
<td>+12V DC Out</td>
<td>+12V DC power supply</td>
</tr>
<tr>
<td>20</td>
<td>Ground</td>
<td>Ground</td>
</tr>
</tbody>
</table>
8.1 mvHYPERION-CLx

IDC multi-pin connector 2 x 10 Pol RM 2.54 x 2.54 mm.

**Note**

Pins are not opto-isolated, feature no EMC filter and are not protected against overload and overvoltage. Digital signals (pins 9-16) are LVTTL signals and not 5V tolerant. Failure to take this into account may result in the destruction of the board.

**Attention**

Without an additional card with corresponding snubbers these signals must not conducted!

8.1.2.8 Switches

<table>
<thead>
<tr>
<th>mvHYPERION</th>
<th>-CLb</th>
<th>-CLe</th>
<th>-CLm</th>
<th>-CLf</th>
</tr>
</thead>
</table>

Switch S1

Flash memory

<table>
<thead>
<tr>
<th>Position</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Def.</td>
<td>Case of need FPGA version is loaded (write protected)</td>
</tr>
<tr>
<td>User</td>
<td>FPGA version, which can be updated, is loaded.</td>
</tr>
</tbody>
</table>

Switch S2

Switch between TTL (5V) and PLC (24V) as well as Trigger and Sync on connector J3

<table>
<thead>
<tr>
<th>Position</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>on</td>
<td>TTL (5V)</td>
</tr>
<tr>
<td>off</td>
<td>PLC (24V)</td>
</tr>
</tbody>
</table>

Switch S3

Switch between TTL (5V) and PLC (24V) as well as Trigger and Sync on connector J4

<table>
<thead>
<tr>
<th>Position</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>on</td>
<td>TTL (5V)</td>
</tr>
<tr>
<td>off</td>
<td>PLC (24V)</td>
</tr>
</tbody>
</table>
8.1.2.9 Digital I/Os

Figure 10: Trigger-In mvHYPERION-CLx
8.1.2.9.1 Opto-isolated digital inputs

**TTL compatible threshold (standard):**

<table>
<thead>
<tr>
<th>VIH, max:</th>
<th>25V</th>
<th>(VIH, max = maximum input voltage, which causes an active signal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIH, min:</td>
<td>4V</td>
<td>(VIH, min = minimum input voltage, which causes an active signal)</td>
</tr>
<tr>
<td>VIH, typ</td>
<td>5V..24V</td>
<td>(VIH, typ = typical input voltage, which causes an active signal)</td>
</tr>
<tr>
<td>VIL, max</td>
<td>1V</td>
<td>(VIL, max = maximum input voltage, which causes an inactive signal)</td>
</tr>
<tr>
<td>VIL, min</td>
<td>-30V</td>
<td>(VIL, min = minimum input voltage, which causes an active signal)</td>
</tr>
<tr>
<td>Ii, max</td>
<td>20mA</td>
<td>(Ii, max = maximum input current)</td>
</tr>
</tbody>
</table>

**PLC compatible threshold (additionally external protective circuit of Z diodes necessary):**

<table>
<thead>
<tr>
<th>VIH, max:</th>
<th>37V</th>
<th>(VIH, max = maximum input voltage, which causes an active signal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIH, min:</td>
<td>15V</td>
<td>(VIH, min = minimum input voltage, which causes an active signal)</td>
</tr>
<tr>
<td>VIL, max</td>
<td>13V</td>
<td>(VIL, max = maximum input voltage, which causes an inactive signal)</td>
</tr>
<tr>
<td>VIL, min</td>
<td>-30V</td>
<td>(VIL, min = minimum input voltage, which causes an active signal)</td>
</tr>
</tbody>
</table>
Max. input frequency of the opto-isolated inputs: 10MHz
An additional series resistor is not necessary at the inputs. The inputs own an internal current limitation and are protected up to -30V against polarity.

8.1.2.9.2 Opto-isolated digital output

**Attention**

The output is not overload protected and doesn’t feature a free-wheeling diode!

There are following conditions:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCEO, max (typ)</td>
<td>35V</td>
</tr>
<tr>
<td>VCEO, max</td>
<td>0.3V</td>
</tr>
<tr>
<td>VCE, sat</td>
<td>1.5V</td>
</tr>
<tr>
<td>IC, max</td>
<td>200mA</td>
</tr>
<tr>
<td>tr, max</td>
<td>5us</td>
</tr>
<tr>
<td>tf, max</td>
<td>2ms, 20us</td>
</tr>
</tbody>
</table>

The output is not protected against overvoltage, overload and polarity reversal.

8.1.2.10 Digital I/Os (J6)

8.1.2.10.1 LVTTL-IN parameters (GPIN3..0)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIH, max..min</td>
<td>3.9V...2.0V</td>
</tr>
<tr>
<td>VIL, max..min</td>
<td>0.8V...-0.3V</td>
</tr>
<tr>
<td>Iin, max</td>
<td>±0.1mA</td>
</tr>
</tbody>
</table>

8.1.2.10.2 LVTTL-OUT parameters (GPOUT3..0 and I2C-SCL)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>OH, min</td>
<td>2.4V</td>
</tr>
<tr>
<td>OL, max</td>
<td>0.4V</td>
</tr>
<tr>
<td>Iout, max</td>
<td>±4mA</td>
</tr>
</tbody>
</table>

8.1.2.10.3 LVTTL-OC parameters (I2C-SDA)
### 8.1.3 Components

<table>
<thead>
<tr>
<th>mvHYPERION-CLx</th>
<th>-CLb</th>
<th>-CLE</th>
<th>-CLm</th>
<th>-CLf</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Video input</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Signal format</td>
<td>CameraLink™ (MiniCL)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Video input</td>
<td>1x BASE</td>
<td>2x BASE or 1x MEDIUM</td>
<td>2x BASE or 1x MEDIUM</td>
<td>1x BASE or 1x MEDIUM or 1x FULL</td>
</tr>
<tr>
<td>Max. CL clock</td>
<td>85 MHz</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supported CL specification</td>
<td>1.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Resolution</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Horizontal / vertical</td>
<td>64 K / not limited</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Pixel formats</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RGB</td>
<td>24 / 30 / 32 bit</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gray</td>
<td>8 / 10 / 12 / 14 / 16 bit</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bus</td>
<td>PCIe Express® x1</td>
<td>PCIe Express® x4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Continuous data rate</td>
<td>Max. 200 MB/s</td>
<td>Max. 620 MB/s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peak data rate</td>
<td>Max. 250 MB/s</td>
<td>Max. 1 GB/s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Payload size</td>
<td>Up to 512 Bytes</td>
<td>Up to 256 Bytes</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Digital in and outputs</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger-In</td>
<td>1, differential, opto-isolated, 5 to 24V</td>
<td>2, differential, opto-isolated, 5 to 24V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Strobe-Out</td>
<td>1, differential, opto-isolated, max. 30V, 100mA</td>
<td>2, differential, opto-isolated, max. 30V, 100mA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sync-In</td>
<td>1, differential, opto-isolated, 5 to 24V</td>
<td>2, differential, opto-isolated, 5 to 24V</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Current consumption</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCIe 3.3V</td>
<td>Max. 1A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCIe 12V</td>
<td>Max. 0.05A + camera power</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Camera supply</td>
<td>Via PCI Express® 12V max. 0.7A fused</td>
<td>Via PCI Express® 12V max. 2A fused</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Via additional floppy power plug up to 2A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Environmental conditions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
8.1.4 Device Feature And Property Lists

8.1.4.1 mvHYPERION-CLm

8.2 mvHYPERION-32R16

8.2.1 Block diagram

The following block diagram shows schematically how the mvHYPERION-32R16 is designed.

![Figure 13: mvHYPERION-32R16 block diagram](link)
8.2.2 Connectors

8.2.2.1 Status LEDs

<table>
<thead>
<tr>
<th>LED</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PCI Express® connection state</td>
<td>Green: No problem with connection</td>
</tr>
<tr>
<td>2</td>
<td>FPGA state</td>
<td>Green: FPGA is loaded</td>
</tr>
</tbody>
</table>

8.2.2.2 Pinning J1 (68-pol connector)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Note</th>
<th>Pin</th>
<th>Signal</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Power 5V</td>
<td></td>
<td>35</td>
<td>GND (G)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>VIDEO_15</td>
<td></td>
<td>36</td>
<td>GND (G)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>VIDEO30</td>
<td>Not used</td>
<td>37</td>
<td>GND (G)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>VIDEO_14</td>
<td></td>
<td>38</td>
<td>GND (G)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>VIDEO29</td>
<td>Not used</td>
<td>39</td>
<td>GND (G)</td>
<td></td>
</tr>
</tbody>
</table>
Manufacturer of the connector: Nexus
Part No.: 32040168R

8.2.2.3 Audio

Both audio connectors (left and right) are Cinch connectors.

8.2.3 Components

<table>
<thead>
<tr>
<th>Video</th>
<th>Input signal</th>
<th>50Hz</th>
<th>60Hz</th>
<th>Number of video inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Interlaced, gray scale</td>
<td>PAL</td>
<td>NTSC</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>Interlaced, color</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mvHYPERION -32R16</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 8.3 mvHYPERION-HD-SDI

#### 8.3.1 Block diagram

The following block diagram shows schematically how the mvHYPERION-HD-SDI is designed.
8.3.2 Connectors

8.3.2.1 Use of J1..J2

<table>
<thead>
<tr>
<th>Connector usage</th>
<th>mvHYPERION-HD-SDI-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>J1</td>
<td>Camera 1 (3G/HD-SDI signal)</td>
</tr>
<tr>
<td>J2</td>
<td>Camera 2 (3G/HD-SDI signal)</td>
</tr>
</tbody>
</table>

Figure 16: mvHYPERION-HD-SDI-2 block diagram

Figure 17: mvHYPERION-HD-SDI-2

Matrix Vision GmbH
8.3.2.2 Pinning J5 (15-pol D-SUB/HD connector)

![Figure 18: J5](image)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Signal direction</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sync-Out 1 (Trilevel, 75Ω)</td>
<td>OUT</td>
<td>1Vss (75Ω)</td>
</tr>
<tr>
<td>2</td>
<td>Sync-Out 2 (Trilevel, 75Ω)</td>
<td>OUT</td>
<td>1Vss (75Ω)</td>
</tr>
<tr>
<td>3</td>
<td>Sync-Out 3 (Trilevel, 75Ω)</td>
<td>OUT</td>
<td>1Vss (75Ω)</td>
</tr>
<tr>
<td>4</td>
<td>ID2</td>
<td>IN/OUT</td>
<td>TTL (open collector)</td>
</tr>
<tr>
<td>5</td>
<td>Digital Ground RS-485</td>
<td>GND</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>Ground Sync-Out 1</td>
<td>GND</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>Ground Sync-Out 2</td>
<td>GND</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td>Ground Sync-Out 3</td>
<td>GND</td>
<td>-</td>
</tr>
<tr>
<td>9</td>
<td>Camera Power Supply</td>
<td>OUT</td>
<td>+5VDC / &gt;=10W (optional +12VDC)</td>
</tr>
<tr>
<td>10</td>
<td>Ground Sync</td>
<td>GND</td>
<td>-</td>
</tr>
<tr>
<td>11</td>
<td>ID0</td>
<td>IN/OUT</td>
<td>TTL (open collector)</td>
</tr>
<tr>
<td>12</td>
<td>RS-485 TRX-</td>
<td>IN/OUT</td>
<td>RS485-</td>
</tr>
<tr>
<td>13</td>
<td>C/HSync-Out</td>
<td>OUT</td>
<td>TTL (push pull)</td>
</tr>
<tr>
<td>14</td>
<td>VSync-Out</td>
<td>OUT</td>
<td>TTL (push pull)</td>
</tr>
<tr>
<td>15</td>
<td>RS-485 TRX+</td>
<td>IN/OUT</td>
<td>RS485+</td>
</tr>
</tbody>
</table>

8.3.2.3 Pinning J6 (internal digital I/Os)

![Figure 19: J6](image)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Signal direction</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NC</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>Ground</td>
<td>GND</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>SCL</td>
<td>IN/OUT</td>
<td>LVTTL</td>
</tr>
<tr>
<td>4</td>
<td>Ground</td>
<td>GND</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>SDA</td>
<td>IN/OUT</td>
<td>LVTTL</td>
</tr>
<tr>
<td>6</td>
<td>Ground</td>
<td>GND</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>+5V power supply</td>
<td>OUT</td>
<td>+5V DC</td>
</tr>
<tr>
<td>8</td>
<td>+3.3V power supply</td>
<td>OUT</td>
<td>+3.3V DC</td>
</tr>
<tr>
<td>12..9</td>
<td>GPIN3..0</td>
<td>IN</td>
<td>LVTTL(3.3V) input. not 5V tolerant!</td>
</tr>
<tr>
<td>16..13</td>
<td>GPOUT3..0</td>
<td>OUT</td>
<td>LVTTL(3.3V) output. not 5V tolerant!</td>
</tr>
<tr>
<td>17</td>
<td>Ground</td>
<td>GND</td>
<td>-</td>
</tr>
<tr>
<td>18</td>
<td>+12V power supply</td>
<td>OUT</td>
<td>+12V DC</td>
</tr>
<tr>
<td>19</td>
<td>+12V power supply</td>
<td>OUT</td>
<td>+12V DC</td>
</tr>
<tr>
<td>20</td>
<td>Ground</td>
<td>GND</td>
<td>-</td>
</tr>
</tbody>
</table>
IDC multi-pin connector 2 x 10 Pol RM 2.54 x 2.54 mm.

**Note**

Pins are not opto-isolated, feature no EMC filter and are not protected against overload and overvoltage. **Digital signals (pins 9-16) are LVTTL signals and not 5V tolerant. Failure to take this into account may result in the destruction of the board.**

**Attention**

Without an additional card with corresponding snubbers these signals must not conducted!

8.3.3 Components

<table>
<thead>
<tr>
<th>supported signal formats</th>
<th>mvHYPERION-HD-SDI-2</th>
</tr>
</thead>
</table>

MATRIX VISION GmbH
Acquisition of 2 independent standard HD-SDI signals or one standard 3G-SDI signal

<table>
<thead>
<tr>
<th>Max. channels</th>
<th>Format</th>
<th>Frequency (fps)</th>
<th>Video timing/data mapping</th>
<th>Physical layer</th>
<th>Standard data</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1080p</td>
<td>23, 24, 25, 29, 97, 98</td>
<td>$S_{\leftrightarrow}$, $M_{\leftrightarrow}$, $PTE_{\leftrightarrow}$, $ST_{\leftrightarrow}$, $274$</td>
<td>$S_{\leftrightarrow}$, $M_{\leftrightarrow}$, $PTE_{\leftrightarrow}$, $ST_{\leftrightarrow}$, $292$</td>
<td>$Y_{\leftrightarrow}$, $U_{\leftrightarrow}$, $V4_{\leftrightarrow}$, $2:2$</td>
<td>(2x10 Bit)</td>
</tr>
<tr>
<td>1</td>
<td>1080p</td>
<td>50, 59, 94, 96, 60</td>
<td>$S_{\leftrightarrow}$, $M_{\leftrightarrow}$, $PTE_{\leftrightarrow}$, $ST_{\leftrightarrow}$, $425$ / Level A</td>
<td>$S_{\leftrightarrow}$, $M_{\leftrightarrow}$, $PTE_{\leftrightarrow}$, $ST_{\leftrightarrow}$, $424$</td>
<td>$Y_{\leftrightarrow}$, $U_{\leftrightarrow}$, $V4_{\leftrightarrow}$, $2:2$</td>
<td>(2x10 Bit)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Only channel 0 supported; Firmware version $\geq 86$ required</td>
<td></td>
</tr>
</tbody>
</table>

| 2            | 720p   | 23, 24, 25, 29, 97, 98 | $S_{\leftrightarrow}$, $M_{\leftrightarrow}$, $PTE_{\leftrightarrow}$, $ST_{\leftrightarrow}$, $296$ | $S_{\leftrightarrow}$, $M_{\leftrightarrow}$, $PTE_{\leftrightarrow}$, $ST_{\leftrightarrow}$, $292$ | $Y_{\leftrightarrow}$, $U_{\leftrightarrow}$, $V4_{\leftrightarrow}$, $2:2$ | (2x10 Bit) |
| 2            | 1080p  | 50, 59, 94, 96, 60 | $S_{\leftrightarrow}$, $M_{\leftrightarrow}$, $PTE_{\leftrightarrow}$, $ST_{\leftrightarrow}$, $274$ | $S_{\leftrightarrow}$, $M_{\leftrightarrow}$, $PTE_{\leftrightarrow}$, $ST_{\leftrightarrow}$, $292$ | $Y_{\leftrightarrow}$, $U_{\leftrightarrow}$, $V4_{\leftrightarrow}$, $2:2$ | (2x10 Bit) |

The host system puts the two fields together to one frame.
## Acquisition of up to 2 non-standard HD/3G-SDI signals

<table>
<thead>
<tr>
<th>Max. channels</th>
<th>Format</th>
<th>Frequency (fps)</th>
<th>Video timing/data mapping</th>
<th>Physical layer</th>
<th>Non-standard data</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1080p</td>
<td>23, 24, 25, 29, 97, 30</td>
<td>S—M—PTE ST 274</td>
<td>S—M—PTE ST 292-1</td>
<td>Raw(2k), Raw(12-bit)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1080p</td>
<td>50, 59, 94, 60</td>
<td>S—M—PTE ST 425 / Level A</td>
<td>S—M—PTE ST 424-1</td>
<td>Raw(Bit)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 Firmware version &gt;= 86 required.</td>
</tr>
<tr>
<td>1</td>
<td>1080p</td>
<td>50, 59, 94, 60</td>
<td>S—M—PTE ST 425 / Level A</td>
<td>S—M—PTE ST 424-1</td>
<td>Raw(2k)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 only channel 0 supported; Firmware version &gt;= 86 required.</td>
</tr>
<tr>
<td>1</td>
<td>1080p</td>
<td>50(100), 59, 94(119), 88, 60(120)</td>
<td>S—M—PTE ST 425 / Level A</td>
<td>S—M—PTE ST 424-1</td>
<td>Raw(2k)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 frames in one double image height; Only channel 0 supported; Firmware version &gt;= 86 required.</td>
</tr>
<tr>
<td>2</td>
<td>720p</td>
<td>23, 24, 25, 29, 97, 30</td>
<td>S—M—PTE ST 296</td>
<td>S—M—PTE ST 292-1</td>
<td>Raw(2k), Raw(12-bit)</td>
<td></td>
</tr>
</tbody>
</table>
### Interface

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bus</td>
<td>PCI Express® x4</td>
</tr>
<tr>
<td>Continuous data rate</td>
<td>Max. 640 MB/s</td>
</tr>
<tr>
<td>Peak data rate</td>
<td>Max. 1 GB/s</td>
</tr>
<tr>
<td>Payload size</td>
<td>Up to 256 Bytes</td>
</tr>
</tbody>
</table>

### Current consumption

<table>
<thead>
<tr>
<th>Voltage</th>
<th>Current (Max.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCIe 3.3V</td>
<td>1A</td>
</tr>
<tr>
<td>PCIe 12V</td>
<td>0.05A + camera power</td>
</tr>
</tbody>
</table>

### Camera supply

<table>
<thead>
<tr>
<th>Method</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Via PCI Express® 12V fused</td>
</tr>
</tbody>
</table>

### Environmental conditions

<table>
<thead>
<tr>
<th>Condition</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambient temperature</td>
<td>0 up to 45 C</td>
</tr>
<tr>
<td>Storage temperature</td>
<td>-20 up to 70 C</td>
</tr>
<tr>
<td>Humidity</td>
<td>10 up to 90 % non-condensing</td>
</tr>
</tbody>
</table>

### Dimensions

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length</td>
<td>155 mm</td>
</tr>
<tr>
<td>Width</td>
<td>111.1 mm</td>
</tr>
</tbody>
</table>
9 Application Usage

9.1 wxPropView

wxPropView (p. 48) is an interactive GUI tool to acquire images and to configure the device and to display and modify the device properties of MATRIX VISION GmbH hardware. After the installation you can find wxPropView (p. 48)

- as an icon with the name "wxPropView" on the desktop (Windows) or
- in "~/mvimpact-acquire/apps/mvPropView/x86" (Linux).

wxPropView - Introduction:

9.1.1 How to work with wxPropView

wxPropView - Working with wxPropView:

9.1.1.1 First View of wxPropView

wxPropView (p. 48) consists of several areas:
Figure 1: wxPropView started

- "Menu Bar"
  (to work with wxPropView (p. 48) using the menu)

- "Upper Tool Bar"
  (to select and initialize a device, acquire images, play a recorder sequence)

- "Left Tool Bar"
  (to hide and show parts of the GUI)

- "Status Tool Bar"

- "Main Window" with
  - "Grid"
    (tree control with the device settings accessible by the user)
  - "Display"
    (for the acquired images)

- "Analysis"
  (information about whole images or an AOI)

By clicking on F1 you will get the HELP dialog.

Now, you can initialize a device by
• selecting it in the drop down list in the "Upper Tool Bar" and

• clicking on "Use".

After having successfully initialized a device the tree control in the lower left part of the "Main Window" will display the properties (settings or parameters) (according to the "interface layout") accessible by the user.

You've also got the possibility to set your "User Experience". According to the chosen experience, the level of visibility is different:

• Beginner (basic camera settings/properties are visible)

• Expert (e.g. all advanced image processing are visible)

• Guru (all settings/properties are visible)

Properties displayed in light grey cannot be modified by the user. Only the properties, which actually have an impact on the resulting image, will be visible. Therefore, certain properties might appear or disappear when modifying another properties.

To permanently commit a modification made with the keyboard the ENTER must be pressed. If leaving the editor before pressing ENTER will restore the old value.

9.1.1.2 How to see the first image

As described earlier, for each recognized device in the system the devices serial number will appear in the drop down menu in the upper left corner of the "Upper Tool Bar". When this is the first time you start the application after the system has been booted this might take some seconds when working with devices that are not connected to the host system via PCI or PCIe.

Once you have selected the device of your choice from the drop down menu click on the "Use" button to open it.

When the device has been opened successfully, the remaining buttons of the dialog will be enabled:

Note

Following screenshots are representative and where made using a mvHYPERION frame grabber as the capturing device.
Now, you can capture an image ("Acquisition Mode": "SingleFrame") or display live images ("Continuous"). Just

- select an "Acquisition Mode" e.g. "SingleFrame" and
- click the "Acquire" button.

**Note**

The techniques behind the image acquisition can be found in the developers sections.

The frame rate depends on

- the camera,
- the pixel clock of the sensor
9.1.1.2.1 Record Mode

It is also possible to record image sequences using wxPropView.

1. For this, you have to set the size of the recorder in "System Settings -> RequestCount" e.g. to 100. This will save the last 100 requests in the request queue of the driver, i.e. the image data including the request info like frame number, time stamp, etc.

2. Afterwards you can start the recording by clicking the Rec. button.

3. With the Next and Prev. buttons you can display the single images.

If you switched on the request info overlay (right-click on the display area and select the entry to activate this feature), these information will be displayed on the image, too. With the timestamp you can see the interval of the single frames in microseconds.

9.1.1.2.2 Hard Disk Recording

You can save acquired images to the hard disk the following way:

1. In the "Menu Bar" click on "Capture -> Recording -> Setup Hard Disk Recording".

2. Confirm with "Yes".

3. Afterwards select the target folder for the images.

4. Finally, choose the file format of the acquired images.

9.1.1.3 Using the analysis plots

With the analysis plots you have the possibility to get image details and to export them (p. 59).

9.1.1.3.1 Spatial noise histogram

The spatial noise histogram calculates and evaluates statistically the difference between two neighbouring pixels in vertical and horizontal direction. I.e. it shows the sensor’s spatial background pattern like the sensitivity shifts of each pixel. An ideal sensor or camera has a spatial noise of zero. However, you have to keep in mind the temporal noise as well.
9.1 wxPropView

Figure 6: wxPropView - Spatial noise histogram

Read: Channel::Direction (Mean difference, most frequent value count/value, Standard deviation)

Example: For a single channel (Mono) image the output of 'C0Hor(3.43, 5086/0, 9.25), C0Ver(3.26, 4840/0, 7.30)
will indicate that the mean difference between pixels in horizontal direction is 3.43, the most frequent difference is 0
and this difference is present 5086 times in the current AOI. The standard deviation in horizontal direction is 9.25.
The C0Ver value list contains the same data but in vertical direction.

9.1.1.3.2 Temporal noise histogram

The temporal noise histogram shows the changes of a pixel from image to image. This method is more stable
because it is relatively independent from the image content. By subtracting two images, the actual structure is elimi-
nated, leaving the change of a pixel from image to image, that is, the noise. When capturing images, all parameters
must be frozen, all automatic mechanisms have to be turned off and the image may not have underexposed or
saturated areas. However, there are no picture signals without temporal noise. Light is a natural signal and the
noise always increases with the signal strength. If the noise only follows the natural limits, then the camera is good.
Only if additional noise is added the camera or the sensor has errors.

Figure 7: wxPropView - Temporal noise histogram
Read: Channel# (Mean difference, most frequent value count/value, Standard deviation)

**Example:** For a single channel (Mono) image the output of 'C0(3.43, 5086/0, 9.25) will indicate that the mean difference between pixels in 2 consecutive images is 3.43, the most frequent difference is 0 and this difference is present 5086 times in the current AOI. The standard deviation between pixels in these 2 images is 9.25. Please note the impact of the 'Update Interval' in this plot: It can be used to define a gap between 2 images to compare. E.g. if the update interval is set to 2, the differences between image 1 and 3, 3 and 5, 5 and 7 etc. will be calculated. In order to get the difference between 2 consecutive images the update interval must be set to 1!

### 9.1.1.4 Storing and restoring settings

When **wxPropView** (p.43) is started for the first time, the values of properties set to their default values will be displayed in green to indicate that these values have not been modified by the user so far. Modified properties (even if the value is the same as the default) will be displayed in black.

![wxPropView - Storing settings](image)

Figure 8: wxPropView - Storing settings

Settings can be stored in several ways (via the "Menu Bar": "Action -> Capture Settings -> Save Active Device Settings"): 

- "As Default Settings For All Devices Belonging To The Same Family (Per User Only)": As the start-up parameters for every device belonging to the same family, e.g. for mvBlueCOUGAR-X, mvBlueCOUGAR-XD.
- "As Default Settings For All Devices Belonging To The Same Family And Product Type": As the start-up parameters for every device belonging to the same product, e.g. for any mvBlueCOUGAR-X but not for mvBlueCOUGAR-XD.
- "As Default Settings For This Device(Serial Number)": As the start-up parameters for the currently selected device.
- "To A File": As an XML file that can be used e.g. to transport a setting from one machine to another or even to use the settings configured for one platform on another (Windows <-> Linux).
During the startup of a device, all these setting possibilities show different behaviors. The differences are described in chapter **Settings behavior during startup** (p. 25).

Restoring of settings previously stored works in a similar way. After a device has been opened the settings will be loaded automatically as described in **Settings behavior during startup** (p. 25).

However, at runtime the user has different load settings possibilities (via the "Menu Bar": "Action -> Capture Settings -> Load Active Device Settings")

- explicitly load the device family specific settings stored on this machine (from "The Default Settings Location For This Devices Family (Per User Only)")
- explicitly load the product specific settings stored on this machine (from "The Default Settings Location For This Devices Family And Product Type")
- explicitly load the device specific settings stored on this machine (from "The Default Settings Location For This Device(Serial Number)")
- explicitly load device family specific settings from a XML file previously created ("From A File")

**Note**

With "Action -> Capture Settings -> Manage..." you can delete the settings which were saved on the system.

![Figure 9: wxPropView - Restoring settings](image)

**9.1.1.5 Properties**

All properties and functions can be displayed in the list control on the lower left side of the dialog. To modify the value of a property select the edit control right of the properties name. Property values, which refer to the default value of the device, are displayed in green. A property value once modified by the user will be displayed in black (even if the value itself has not changed). To restore its default value of a single property

- right click on the name of the property and
- select "Restore Default".

MATRIX VISION GmbH
To restore the default value for a complete list (which might include sub-lists)

- right click on the name of a list and
- select "Restore Default".

In this case a popup window will be opened and you have to confirm again.

![Figure 10: wxPropView - Restore the default value of a property](image)

Most properties store one value only, thus they will appear as a single entry in the property grid. However, properties are capable of storing more than one value, if this is desired. A property storing more than one value will appear as a parent list item with a **WHITE** background color (lists will be displayed with a grey background) and as many child elements as values stored by the property. The **PARENT** grid control will display the number of values stored by the property, every child element will display its corresponding value index.

If supported by the property, the user might increase or decrease the number of values stored by right clicking on the **PARENT** grid element. If the property allows the modification the pop up menu will contain additional entries now:

![Figure 11: wxPropView - A resizable property](image)

When a new value has been created it will be displayed as a new child item of the parent grid item:
Currently, only the last value can be removed via the GUI and a value can't be removed, when a property stores one value only.

Also the user might want to set all (or a certain range of) values for properties that store multiple values with a single operation. If supported by the property, this can also be achieved by right clicking on the PARENT grid element. If the property allows this modification the pop up menu will again contain additional entries:

![Figure 12: wxPropView - A resized property](image)

![Figure 13: wxPropView - Setting multiple property values](image)
It's possible to either set all (or a range of) elements of the property to a certain value OR to define a value range, that then will be applied to the range of property elements selected by the user. The following example will explain how this works:

In this sample the entries 0 to 255 of the property will be assigned the value range of 0 to 255. This will result in the following values AFTER applying the values:

![Image](image-url)

**Figure 14:** wxPropView - Setting multiple property values within a certain value range

**Figure 15:** wxPropView - After applying the value range to a property
9.1.1.6 Methods

Method appears as entries in the tree control as well. However, their name and behavior differs significantly from the behavior of properties. The names of method objects will appear in ‘C’ syntax like e.g. “int function( char*, int )”. This will specific a function returning an integer value and expecting a string and an integer as input parameters. To execute a method object:

- right click on the name of a method and
- select "Execute" from the popup menu:

![Figure 16: wxPropView - Calling a method object](image)

Parameters can be passed to methods by selecting the edit control left of a method object. Separate the parameters by blanks. So to call a function expecting a string and an integer value you e.g. might enter "testString 0" into the edit control left of the method.

The return value (in almost every case an error code as an integer) will be displayed in the lower right corner of the tree control. The values displayed here directly correspond the error codes defined in the interface reference and therefore will be of type TDMR_ERROR or TPROPHANDLING_ERROR.

9.1.1.7 Copy grid data to the clipboard

Since wxPropView (p.48) version 1.11.0 it is possible to copy analysis data to the clipboard. The data will be copied in CSV style thus can be pasted directly into tools like Open Office™ or Microsoft® Office™.

Just

- right-click on the specific analysis grid when in numerical display mode and
- select "Copy grid to clipboard" from the pop up menu.
9.1.1.8 Import and Export images

wxPropView (p. 48) offers a wide range of image formats that can be used for exporting captured image to a file. Some formats e.g. like packed YUV 4:2:2 with 10 bit per component are rather special thus they can't be stored into a file like e.g. offered by the BMP file header. When a file is stored in a format, that does not support this data type wxPropView (p. 48) will convert this image into something that matches the original image format as close as possible. This, however, can result in the loss of data. In order to allow the storage of the complete information contained in a captured image wxPropView (p. 48) allows to store the data in a raw format as well. This file format will just contain a binary dump of the image with no leader or header information. However, the file name will automatically be extended by information about the image to allow the restoring of the data at a later time.

All image formats, that can be exported can also be imported again. Importing a file can be done in 3 different ways:

- via the menu (via the "Menu Bar": "Action -> Load image...")
- by dragging an image file into an image display within wxPropView (p. 48)
- by starting wxPropView (p. 48) from the command line passing the file to open as a command line parameter (p. 94) (under Windows® e.g. "wxPropView.exe MyImage.png" followed by [ENTER])

When importing a "*.raw" image file a small dialog will pop up allowing the user to define the dimensions and the pixel format of the image. When the file name has been generated using the image storage function offered by wxPropView (p. 48), the file name will be passed and the extracted information will automatically be set in the dialog thus the user simply needs to confirm this information is correct.
9.1.1.9 Setting up multiple display support and/or work with several capture settings in parallel

`wxPropView` (p. 48) is capable of

- dealing with multiple capture settings or acquisition sequences for a single device and in addition to that
- it can be configured to deal with multiple image displays.

For frame grabbers with multiple input channels this e.g. can be used to display live images from all input channels simultaneously. This even works if each input channel is connected to a different video signal in terms of resolution and timing.

The amount of parallel image displays can be configured via the command line parameter (p. 94) "dcx" and "dcy". In this step by step setup `wxPropView` (p. 48) has been started like this from the command line:

```
wxPropView dcx=1 dcy=2
```

This will result in 1 display in horizontal direction and 2 in vertical direction.

Since

`mvIMPACT Acquire 2.18.1`

It is also possible to change the amount of display at runtime via "Settings -> Image Displays -> Configure Image Display Count".
Additional capture settings can be created via "Menu Bar": "Capture -> Capture Settings -> Create Capture Settings". The property grid will display these capture settings either in "Developers" or in "Multiple Settings View".

Now, in order to set up wxPropView (p. 48) to work with 2 instead of one capture setting,

1. Various additional capture setting can be created. In order to understand what a capture setting actually is please refer to
   - "Working with settings" chapter of the "mvIMPACT Acquire API" manuals.

Creating a capture setting is done via "Capture -> Capture Settings -> Create Capture Setting".
2. Then, the user is asked for the name of the new setting.

![Image of wxPropView - Create capture setting - Choosing name](image1.png)

Figure 21: wxPropView - Create capture setting - Choosing name

3. And finally for the base this new setting shall be derived from.

![Image of wxPropView - Create capture setting - Choosing base](image2.png)

Figure 22: wxPropView - Create capture setting - Choosing base

Afterwards, in this example we end up having 2 capture settings:

- a "Base" setting, which is always available
- a "NewSetting1", which has been derived from "Base".
As "NewSetting1" has been derived from "Base" changing a property in "Base" will automatically change this property in "NewSetting1" if this property has not already been modified in "NewSetting1". Again to get an understanding for this behaviour please refer to

- "Working with settings" chapter of the "mvIMPACT Acquire API" manuals.

Now, to set up wxPropView (p. 48) to display all images taken using capture setting "Base" in one display and all image taken using capture setting "NewSetting1" in another display the capture settings need to be assigned to image displays via "Capture -> Capture Settings -> Assign To Display(s)".
By default a new setting when created will be assigned to one of the available displays in a round-robin scheme, thus when there are 3 displays, the first (Base) setting will be assigned to "Display 0", the next to "Display 1", the next to "Display 2" and a fourth setting will be assigned to "Display 0" again. The setting to display relationships can be customized via "Capture -> Capture Settings -> Assign to Display(s)".

As each image display keeps a reference to the request, this image belongs to the driver can’t re-use the request buffer until a new request is blitted into this display. Thus, it might be necessary to increase the number of request objects the driver is working with if a larger number of displays are involved. The minimum number of requests needed is 2 times the amount of images displays. The number of requests used by the driver can be set up in the drivers property tree:
Finally, *wxPropView* (p. 48) must be configured in order to use all available capture settings in a round-robin scheme. This can be done by setting the capture setting usage mode to **Automatic** via "Capture -> Capture Settings -> Usage Mode".

That's it. Now, starting a live acquisition will display live images in both displays and each display is using a different set of capture parameters. If a device supports parallel acquisition from multiple input channels, this will increase

- the used bandwidth and also
- the CPU load
as **wxPropView** now needs to display more images per second. Each display can be configured independently thus e.g. one display can be used scaled while the other displays 1:1 data. The analysis plots can be assigned to a specific display by left-clicking on the corresponding image display, the info plot will plot a graph for each capture setting in parallel.

![wxPropView](image)

Figure 28: wxPropView - Running example

When only one setting shall be used at a given time, this can be achieved by setting the capture setting usage mode back to "Manual" via "Capture -> Capture Settings -> Usage Mode". Then the setting that shall be used can be manually selected in the request control list:
This can even be changed during a running acquisition.

9.1.1.10 Bit-shifting an image

wxPropView (p.48) shows snapped or live images in the display area of the GUI. The area, however, shows the most significant bits (msb) of the image in the 8 bit display.

The following image shows how a mid-grey 12 bit pixel of an image is displayed with 8 bit. Additionally, two shifts are shown.

**Mono12 (12 bit):**

```
   0   1   1   1   1   0   0   1   1   1   1   0   0
```

**Display (8 bit):**

```
   0   1   1   1   1   0   0   1   1   1   1   0   0
   0   7
```

**Display (8 bit); Shift: 1:**

```
   0   1   1   1   1   0   0   1   1   1   1   0   0
   0   7
```

**Display (8 bit); Shift: 2:**

```
   0   1   1   1   1   0   0   1   1   1   1   0   0
   0   7
```
In this particular case, the pixel will be brighter (as the most significant bits are 1’s). Perhaps you already recognized it. Each shift means that each pixel value is multiplied or divided by 2 according to the direction.

Anyway, there is one restriction in the 8 bit display:

If the pixel value is greater than 255, the pixel value will be clipped to 255. To describe this from a programmer’s view; \( a \) represents the pixel value:

\[
a = \begin{cases} 
255 & \text{if } a > 255 \\
 a & \text{otherwise}
\end{cases}
\]

With \textit{wxPropView} you can shift the bits in the display using the left and right arrow keys. Furthermore you can turn on the monitor display to compare the images synchronously.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{wxPropView_Bit-shifting.png}
\caption{wxPropView - Bit-shifting an Image: https://www.matrix-vision.com/tl_files/mv11/trainings/wxPropView/wxPropView_Bit-shifting/index.html}
\end{figure}

9.1.1.11 Changing the view of the property grid to assist writing code that shall locate driver features

With \textit{wxPropView} it is possible to switch the views between “Standard View” (user-friendly) and “Developers View”. While the first (default) view will display the device drivers feature tree in a way that might be suitable for most users of a GUI application it might present the features in a slightly different order as they actually are implemented in the device driver. The developers view switches the tree layout of the application to reflect the feature tree exactly like it is implemented an presented by the SDK. It can be helpful when writing code that shall locate a certain property in the feature tree of the driver using the C, C++, or .NET interface. The feature hierarchy displayed here can directly be used for searching for the features using the “ComponentLocator (C++/.NET)” objects or “DMR_FindList (C)” and “OBJ_GetHandleEx (C)” functions.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{DevelopersView.png}
\caption{Developers View}
\end{figure}
9.1.12 Accessing log files

Since mvIMPACT Acquire 2.11.9

Using Windows, it is possible to access the log files generated by MATRIX VISION via the Help menu. Sending us the log files will speed up support cases.

![wxPropView - Help menu](image)

The options are to

- directly open the logs folder, to
- create a zip file with all the logs, and to
- open the systems default email client to send an email to support@matrix-vision.com.

9.1.2 How to configure a device

As described above, after the device has been initialized successfully in the “Grid” area of the GUI the available properties according to the chosen "interface layout" (e.g. GenICam) are displayed in a hierarchy tree.

![wxPropView - Configuring a device](image)

The next chapter will show how to set the interface layout and which interface you should use according to your needs.
9.1.2.1 Different interface layouts

Devices belonging to this family only support the **Device Specific** interface layout which is the common interface layout supported by most MATRIX VISION devices.

**GenICam** compliant devices can be operated in different interface layouts. Have a look at a GenICam compliant device for additional information.

9.1.2.2 Configuring different trigger modes

To configure a device for a triggered acquisition, in **wxPropView** (p. 48) the property "Image Setting -> Camera -> TriggerMode" ("DeviceSpecific interface layout") or "Setting -> Base -> Camera -> GenICam -> Acquisition Control -> Trigger Selector" ("GenICam interface layout") is available.

9.1.2.3 Testing the digital inputs

**Note**

The following description will be significant if you are using the "DeviceSpecific interface layout". In GenICam layout, the "Digital I/O" section can be found in "Setting -> Base -> Camera -> GenICam -> Digital I/O Control".

For performance reasons, device drivers will not automatically update their digital input properties if nobody is interested in the current state. Therefore, in order to check the current state of a certain digital input, it is necessary to manually refresh the state of the properties. To do this please right-click on the property you are interested in and select "Force Refresh" from the pop-up menu.

**GenICam interface layout only:**

Some devices might also offer an event notification if a certain digital input changed its state. This event can then be enabled

- via the "EventSelector" in "Setting -> Base -> Camera -> GenICam -> Event Control".
- Afterwards, a callback can be registered by right-clicking on the property you are interested in again.
- Now, select "Attach Callback" from the pop-up menu and switch to the "Output" tab in the lower right section of wxPropView (Analysis tabs).

Whenever an event is send by the device that updates one of the properties a callback has been attached to, the output window will print a message with some information about the detected change.
9.1.2.4 Working with camera descriptions

Certain capture device (e.g. frame grabber) can process data from a wide range of imaging devices (e.g. cameras). However, in order to interpret the incoming data from an imaging device correctly, the capture device needs to be given a certain amount of information about the structure of the video signal.

The "mvIMPACT Acquire" interface addresses this necessity by the introduction of so called "camera descriptions". A "camera description" is a certain set of parameters that should enable the capture device to cope with the incoming image data to reconstruct a correct image from the imaging device in the memory of the host system. For instance, this information may contain information whether the image is transmitted as a whole or if it’s transmitted as individual blocks (e.g. when dealing with interlaced cameras) that need to be reconstructed in a certain way to form the complete image.

Each capture device will support different sets of parameters. For example some capture devices will only be able to capture image data from standard video source such as a PAL or NTSC compliant camera, while others might only be capable to acquire data from digital image source such as CameraLink® compliant cameras. To reflect these device specific capabilities "camera descriptions" have been grouped into different base classes. See e.g. `mvIMPACT::acquire::CameraDescriptionStandard` to find out how the basic structure of these objects look. Which basic "camera description" classes are supported by an individual device can be seen directly after the device has been initialised by looking in the "camera description" list. By default this list will contain one description for each supported basic family:
To select a certain camera description to be used to prepare the capture device for the expected data the property "Type" under "Image Settings -> Camera" can be modified. Here every available set of camera parameters will be listed:
Now, when a camera is connected, that differs in one or more parameters from the default offered by one of the available base classes and no special description for the imaging device in question is available a new matching description must be generated.

**Note**

It's also possible to modify one of the standard descriptions to adapt the parameter set to the used imaging device, but this method is not recommend as this would define something to be "standard", which in fact is not. Therefore it is not possible to store the standard descriptions permanently. It is, however, possible to modify and work with the changed parameters, but these changes will be lost once the device is closed.

The recommended way of adapting an imaging source to a capture device is to create a new description for a imaging device that does not completely fall into one of the offered standard descriptions. The first thing to decide when creating a new camera description is to which existing description offers the closest match for the new description. Once this has been decided a copy of this description can be created with an arbitrary name (that must be unique within the family the description is created from). Under wxPropView (p. 48) this can be achieved by

- typing the new name in the parameter edit control right of the “Copy” method of the camera description to create the copy from.
- Afterwards, press ENTER to commit the new name and then the “Copy” method can be invoked
- by right clicking on the name of the function and
- selecting “Call” from the popup menu:

![wxPropView - Creating a new camera description](image)
Afterwards, the newly created camera description will be added to the list of existing ones. Its parameters at this point will match the parent description (the one the “Copy” method was executed from) completely.

Now, the reason for creating a new camera description was that the parameters in the existing description didn’t exactly match the connected imaging device. Therefore, the next step would probably be to modify some of the parameters. Once this has been done (or before) the newly created description can be selected via the property “Type” under “Image Settings -> Camera”.

Figure 33: wxPropView - The newly created camera description
Figure 34: wxPropView - Selecting the newly created camera description

**Note**

A new camera description will **NOT** be stored permanently by default. In order to make this description available the next time the capture device is initialised, the newly created description must be exported via a function call.

To store a camera description permanently the "Export" method of the new camera description must be invoked. The method does not require any parameters so it can be executed directly by right clicking on the name of the function and selecting "Execute" from the popup menu:
As a direct result the modified settings will become the new default values of this particular camera description. **wxPropView** (p. 48) indicates this by displaying all values belonging to the description in green now:

![Diagram of wxPropView](image)

**Figure 35: wxPropView - Exporting a created camera description**
Figure 36: wxPropView - After exporting a new camera description

Note

Again please note, that this will NOT work for one of the standard camera descriptions. Whenever the user tries to export one of these, the error DMR_EXECUTION_PROHIBITED will be returned.

When exporting a camera description a file in XML format will be written to disc. Under Windows® camera descriptions will be stored under "%ALLUSERS%\Documents\MATRIX VISION\mvIMPACT acquire\CameraFiles" or "%MVIMPACT_ACQUIRE_DATA_DIR%\CameraFiles" which will point to the same folder. Under Linux® this directory will be "*/etc/matrix-vision/mvimpact-acquire/camerafiles" while under other platforms these files will end up in the current working directory.

Now, when closing and re-opening a device only the default camera descriptions and the one selected before settings have been saved will appear in the list of camera descriptions. This is to save memory. However, all detected camera descriptions will be available via the property "Type" under "Image Settings -> Camera":

MATRIX VISION GmbH
Figure 37: wxPropView - After re-opening of the device

Once a description is selected, that hasn't been in the list of camera descriptions before, it will be created and thus will become available for modifications again:
Again: For a different camera a new description should be generated, to operate complex cameras in different modes, a either a new description can be generated or an existing one can be modified.

After a camera has been modified the "Import" method can be used to fall back to the values stored in the camera description file:
### Figure 39: wxPropView - Invoking the "Import" command of a camera description

This will restore the default settings for this description:
9.1.2.4.1 Configuring an unknown camera

If you need a camera description (p.72) of an unknown CameraLink or SDI camera, wxPropView (p.48) supports you with three properties, which can be found in "Info -> Camera":

- DataCycleCounterLine0
- DataCycleCounterLine1
- LineCounter

For line scan cameras, the property DataCycleCounterLine0 is enough to know.
For area scan cameras, you will need all three properties.
You can take the information in "Info -> Camera" to enter the values in "Camera Descriptions". (The figures 37 and 38 are showing CameraLink examples with default values in "Camera Descriptions". The values from "Info -> Camera" are not entered yet.)

**Note**

To get the current values of the properties mentioned above you have to "Acquire" a "SingleFrame" first!
9.1.2.5 Basic trigger techniques in CameraLink systems

9.1.2.5.1 Area scan cameras

"Mode 1: Frame grabber is triggered, free running camera"

![Diagram of Frame grabber triggered, free running camera](image1)

Figure 43: Frame grabber is triggered, free running camera

"Mode 2: Camera is triggered"

![Diagram of Camera triggered](image2)

Figure 42: Camera is triggered
9.1.2.5.2 Line scan cameras

"Mode 1: Camera is triggered by frame grabber"

Figure 45: Camera is triggered by frame grabber

"Mode 2: External trigger signal triggers camera"

Figure 46: External trigger signal triggers camera
9.1.2.6 Triggering with mvHYPERION

9.1.2.6.1 Area scan cameras

"Mode 1"

In this mode, there is no change in the "Digital I/O" interface necessary.

Now, please follow these steps to run Mode 1 with mvHYPERION:

1. In "Image Setting -> Camera -> TriggerControls -> Frame Start" set "TriggerMode" to "On".
2. Choose the "TriggerSource" input (normally "Trigger-In").
3. Choose the "TriggerActivation" according to the application (e.g. "FallingEdge", "RisingEdge", etc.).

In order to that the camera will send an image stream continuously and the next image after a trigger event will be acquired.

"Mode 2"

In this mode, there is no setup in "TriggerControls" necessary ("TriggerMode" = "Off").

Now, please follow these steps to run Mode 2 with mvHYPERION:

1. In "Digital I/O" set "ControlMode" to "PulseStartConfiguration".
2. Now, set the wanted mode of the used CameraLink signal (normally "DigitalOutputs -> CC1"), which is used for triggering "TriggerSource" input (normally "Trigger-In"):
   (a) "SinglePulse": On the basis of "PulseStartConfiguration" a signal is created and given to the camera for triggering. Delay time and pulse width can be defined.
   (b) "PassThrough": The signal of the chosen input will be negated in timing and pulse width or not passed to the camera.
3. In "PulseStartConfiguration" set the PulseStartTrigger to "DigitalSignal":
   (a) Set "DigitalSignal" to the wished input (normally: "Trigger-In").
   (b) If the signal happens too fast, you can divide the trigger frequency with the "TriggerDivider".
   (c) "TriggerMoment" shows the starting time with falling or rising edge.

Now, after a external trigger signal on the trigger input of the frame grabber, a trigger signal is generated on the CameraLink connection ("CC1") and passed to the camera. In order to that, the camera acquires an image and sends it to the frame grabber. You do not need a trigger on the frame grabber for the image acquisition given that the frame grabber waits for the next image.
9.1.2.6.2 Line scan cameras

"Mode 1"

In this mode, there is no setup in "TriggerControls" necessary ("TriggerMode" = "Off").

Now, please follow these steps to run Mode 1 with mvHYPERION:

1. In "Digital I/O" set the mode to "SinglePulse" of the used CameraLink signal (normally "DigitalOutputs -> CC1"), which is used for triggering and
   (a) define "Priority", "Delay_us" and "Width_us" (pulse width).
   (b) Afterwards, set the "PulseStartConfiguration" to "PulseStartConfiguration0".
2. In "PulseStartConfiguration"
   (a) set the PulseStartTrigger to "Periodically".
   (b) The Property "Frequency_Hz" defines the line frequency, which is passed to the camera over "CC1".

"Mode 2"

"TriggerControls" settings in this mode are optionally.

Following settings in "Digital I/O" are necessary:

1. Now, set the wanted mode of the used CameraLink signal (normally "DigitalOutputs -> CC1"), which is used for triggering "TriggerSource" input (normally "Trigger-In"):
   (a) "SinglePulse": On the basis of "PulseStartConfiguration" a signal is created and given to the camera for triggering. Delay time and pulse width can be defined.
   (b) "PassThrough": The signal of the chosen input will be negated in timing and pulse width or not passed to the camera. Please choose "Sync-In" as input. No further settings are necessary.
2. In "PulseStartConfiguration" set the PulseStartTrigger to "DigitalSignal":
   (a) Set "DigitalSignal" to the wished input (normally: "Sync-In").
   (b) If the signal happens too fast, you can divide the trigger frequency with the "TriggerDivider".
   (c) "TriggerMoment" shows the starting time with falling or rising edge.

Now, after a external trigger signal on the sync input of the frame grabber, a trigger signal is generated on the CameraLink connection ("CC1") and passed to the camera. In order to that, the camera acquires an image and sends it to the frame grabber. It is possible to trigger the image acquisition of the frame grabber externally. For this, please do following:

1. In "Image Settings -> Camera -> TriggerControls -> Frame Start" set "TriggerMode" to "On".
2. Choose the "TriggerSource" input (normally "Trigger-In").
3. Choose the "TriggerActivation" according to the application (e.g. "FallingEdge", "RisingEdge", etc.).

Per image and frame trigger, the number of lines will be acquired, which were defined in the camera description and the AOI setting before.
9.1.2.7 Camera acquisition techniques

There are different camera acquisition techniques. How you can set them with wxPropView (p.48), which will be shown in the following section.

9.1.2.7.1 StartTrigger

Directly after the trigger signal the acquisition starts. If you have, for example, a line scan camera and want to acquire 1000 lines, 1000 lines will be acquired. During this time, further trigger signals are ignored.

To use StartTrigger, in wxPropView (p.48) you have to

- set "Image Settings -> Camera -> TriggerControls -> FrameStart" to "On",
- select in "FrameStart" the used "TriggerSource" and
- set "Image Settings -> Camera -> TriggerControls -> FrameStop" to "Off".
9.1.2.7.2 TriggerStartStop

In TriggerStartStop there are two trigger sources, one to start the acquisition and the second trigger event to stop it. Between start and stop, there is at least one line pause. The image height is affected by the stop event.
To use **TriggerStartStop**, in **wxPropView** (p. 48) you have to

- set "Image Settings -> Camera -> TriggerControls -> FrameStart" to "On",
- select in "FrameStart" the used "TriggerSource",
- set "Image Settings -> Camera -> TriggerControls -> FrameStop" to "On" and
- select in "FrameStop" the used "TriggerSource".

![screenshot]

**Figure 50: wxPropView - Setting TriggerStartStop**

### 9.1.2.7.3 TriggerStartStop - Restart

With "**TriggerStartStop - Restart**" the first trigger starts the acquisition and following trigger signal stops the previous acquisition and starts the next one.
To use "TriggerStartStop - Restart", in wxPropView (p.48) you have to

- set the same parameters in "Image Settings -> Camera -> TriggerControls -> FrameStart" and "Image Settings -> Camera -> TriggerControls -> FrameStop".
Note

The image height depends on the trigger frequency.

1. trigger period > frame period -> image height complete

2. trigger period < frame period -> image height reduced (line synchronous; next request starts immediately without loss of lines)

9.1.2.7.4 TriggerDelay

With TriggerDelay it is possible to specify a delay after the trigger start event.

![TriggerDelay Diagram]

To use TriggerDelay, in wxPropView (p. 43) you have to

- set in "CameraDescriptions" the "Y" parameter of "ActiveVideoAoi",

Using line scan cameras, the "Y" position specifies the trigger acquisition delay in lines.
9.1.3 Command-line options

It is possible to start wxPropView via command line and controlling the starting behavior using parameters. The supported parameter are as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width or w</td>
<td>Defines the startup width of wxPropView. Example: width=640</td>
</tr>
<tr>
<td>height or h</td>
<td>Defines the startup height of wxPropView. Example: height=460</td>
</tr>
<tr>
<td>xpos or x</td>
<td>Defines the startup x position of wxPropView.</td>
</tr>
<tr>
<td>ypos or y</td>
<td>Defines the startup y position of wxPropView.</td>
</tr>
<tr>
<td>splitterRatio</td>
<td>Defines the startup ratio of the position of the property grids splitter.</td>
</tr>
<tr>
<td></td>
<td>Values between &gt; 0 and &lt; 1 are valid. Example: splitterRatio=0.5</td>
</tr>
<tr>
<td>propgridwidth or pgw</td>
<td>Defines the startup width of the property grid.</td>
</tr>
<tr>
<td>debuginfo or di</td>
<td>Will display debug information in the property grid.</td>
</tr>
<tr>
<td>dic</td>
<td>Will display invisible (currently shadowed) components in the property grid.</td>
</tr>
<tr>
<td>displayCountX or dcx</td>
<td>Defines the number of images displayed in horizontal direction.</td>
</tr>
<tr>
<td>displayCountY or dcy</td>
<td>Defines the number of images displayed in vertical direction.</td>
</tr>
<tr>
<td>fulltree or ft</td>
<td>Will display the complete property tree (including the data not meant to be</td>
</tr>
<tr>
<td></td>
<td>accessed by the user) in the property grid. Example (Tree will be shown)--</td>
</tr>
<tr>
<td></td>
<td>:fulltree=1</td>
</tr>
<tr>
<td>device or d</td>
<td>Will directly open a device with a particular serial number. * will take</td>
</tr>
<tr>
<td></td>
<td>the first device. Example: d=GX000735</td>
</tr>
</tbody>
</table>
9.2 mvDeviceConfigure

mvDeviceConfigure (p. 95) is an interactive GUI tool to configure MATRIX VISION devices. It shows all connected devices.

Various things can also be done without user interaction (e.g., updating the firmware of a device). To find out how to do this please start mvDeviceConfigure and have a look at the available command line options presented in the text window in the lower section (the text control) of the application.

9.2.1 How to set the device ID

The device ID is used to identify the devices with a self defined ID. The default ID on the device’s EEPROM is “0”. If the user hasn’t assigned unique device IDs to his devices, the serial number can be used to select a certain device instead. However, certain third-party drivers and interface libraries might rely on these IDs to be set up in a certain way and in most of the cases this means, that each device needs to have a unique ID assigned and stored in the devices non-volatile memory. So after installing the device driver and connecting the devices setting up these IDs might be a good idea.

To set the ID please start the mvDeviceConfigure (p. 95) tool. You will see the following window:

### Table: QSW and Live Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>qsw</td>
<td>Will forcefully hide or show the Quick Setup Wizard, regardless of the default settings. Example (Quick Setup Wizard will be shown): <code>qsw=1</code></td>
</tr>
<tr>
<td>live</td>
<td>Will directly start live acquisition from the device opened via <code>device</code> or <code>d</code> directly. Example (will start the live acquisition): <code>live=1</code></td>
</tr>
</tbody>
</table>

9.1.3.1 Sample (Windows)

`wxPropView.exe d=* fulltree=1 qsw=0`

This will start the first available device, will hide the Quick Setup Wizard, and will display the complete property tree.

---

MATRIX VISION GmbH
Whenever there is a device that shares its ID with at least one other device belonging to the same device family, \texttt{mvDeviceConfigure} (p. 95) will display a warning like in the following image, showing in this example two \texttt{mvBlueFOX} cameras with an ID conflict:

![Figure 53:mvDeviceConfigure - Conflicting device IDs](image)

**Figure 53:mvDeviceConfigure - Conflicting device IDs**

### 9.2.1.1 Step 1: Device Selection

Select the device you want to set up from the list box.

### 9.2.1.2 Step 2: Open dialog to set the ID

With the device selected, select the menu item \texttt{Action} and click on \texttt{Set ID}.

**Note**

It is also possible to select the action with a right click on the device.

![Figure 54:mvDeviceConfigure - Select action](image)
9.2.1.3 Step 3: Assign the new ID

Enter the new ID and click OK.

Figure 55: mvDeviceConfigure - New ID

Now the overview shows you the list with all devices as well as the new ID. In case there has been an ID conflict before that has been resolved now mvDeviceConfigure (p. 95) will no longer highlight the conflict now:

Figure 56: mvDeviceConfigure - Resolved ID conflict

9.2.2 How to update the firmware

With the mvDeviceConfigure tool it is also possible to update the firmware. In the device list, new firmware versions, if available, will be marked in blue.

These steps are necessary:
9.2.2.1 Step 1: Device Selection

Select the mvHYPERION you want to update from the list box.

**Note**

To update mvHYPERION you have to set "Switch S1" to "User", which is the condition as supplied to customer.
Please have a look at Switches (p. 33) (Connectors (p. 28)), where to find the switch.

9.2.2.2 Step 2: Open dialog to update the firmware

With the device selected, select the menu item *Action* and click on *Update firmware*.

**Note**

It is also possible to select the action with a right click on the device.

![Figure 57: mvDeviceConfigure - Select action](image)

9.2.2.3 Step 3: Firmware file selection

Select the firmware file and open it.

![Figure 58: mvDeviceConfigure - Select firmware file](image)
You will see an information dialog, which says that the firmware will take some time. Please click OK.

Figure 59: mvDeviceConfigure - Select firmware file

Figure 60: mvDeviceConfigure

9.2.2.4 Step 4: Reboot system

Please reboot the system (cold starting).

After the reboot, you can also see via the log window that the update was successfully.
Note

The firmware update is only necessary in some special cases (e.g., to benefit from a new functionality added to the firmware or to fix a firmware related bug). Before updating the firmware be sure what you are doing and have a look into the change log (versionInfo.txt and/or the manual to see if the update will fix your problem).

9.2.3 How to recover a broken firmware update

If something goes wrong during the flash update and the mvHYPERION does not work, you have to accomplish following emergency procedure:

• Please change "Switch S1" to "Def.",
  Please have a look at Switches (p. 33) (Connectors (p. 28)), where to find the switch.
• Reboot the system (cold starting).
• After reboot and during operation set "Switch S1" to "User".
  Please have a look at Technical data (p. 28), where to find the switch.
• Update the firmware (p. 97).

After reboot of the system (cold starting), the new version will be used.

Note

The flash update will take some time and needs processing power. During update mvDeviceConfigure (p. 95) does not respond to user input.

9.2.4 How to allocate image memory

Generally (and used by default), the (image) memory management of the used operating system is sufficient to work with. Sometimes, especially when using large images, it is necessary to define permanent image memory manually.

For this,

1. click with the right mouse button on the device you want to define permanent image memory:

![Configuration tool for MATRIX VISION GmbH devices](image)

Figure 61: mvDeviceConfigure - Image memory
A dialog will be opened.

2. Entered your preferred image memory size.

3. Afterwards, reboot you system.

4. Now, open wxPropView (p.49). In wxPropView you will have a new Property named "Image Memory Manager".

5. Here you have to possible modes:

   (a) With **Automatic**, the driver makes the memory management.

   (b) With **UsePool**, you can define for example the block sizes of the memory.

---

9.2.5 How to disable CPU sleep states a.k.a. C states (< Windows 8)

Modern PC’s, notebook’s, etc. try to save energy by using a smart power management. For this several hardware manufacturers specified the ACPI standard. The standard defines several power states. For example, if processor load is not needed the processor changes to a power saving (sleep) state automatically and vice versa. Every state change will stop the processor for microseconds. This time is enough to cause image error counts!

See also


To disable the power management on the processor level (so-called "C states"), you can use mvDeviceConfigure:  

---

MATRIX VISION GmbH
With Windows XP it is only possible to disable the C2 and C3 states. With Windows Vista / 7 / 8 all C states (1,2, and 3) will be disabled.

**Warning**

Please be sure you know what you do! To turn off the processor's sleep states will lead to a higher power consumption of your system. Some processor vendors might state that turning off the sleep states will result in the processors warranty will expire.

**Note**

Modifying the sleep states using `mvDeviceConfigure` does only affects the current power scheme. For notebooks this will e.g. make a difference depending on whether the notebook is running on battery or not. E.g. if the sleep states have been disabled while running on battery and then the system is connected to an external power supply, the sleep states might be active again. Thus in order to permanently disable the sleep states, this needs to be done for all power schemes that will be used when operating devices.

1. Start `mvDeviceConfigure`.
2. Go to tab "Settings" and unselect "CPU Idle States Enabled".

![Configuration tool for MATRIX VISION GmbH devices](image)

**Figure 64: mvDeviceConfigure - Settings**
The sleep states can also be enabled or disabled from a script by calling `mvDeviceConfigure` like this:

```bash
mvDeviceConfigure.exe set_processor_idle_states=1 quit
```

or

```bash
mvDeviceConfigure.exe set_processor_idle_states=0 quit
```

The additional `quit` will result in the application to terminate after the new value has been applied.

**Note**

With Windows Vista or newer `mvDeviceConfigure` must be started from a command shell with administrator privileges in order to modify the processors sleep states.

### 9.2.6 Command-line options

It is possible to start `mvDeviceConfigure` via command line and controlling the starting behavior using parameters. The supported parameter are as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>setid</code> or <code>id</code></td>
<td>Updates the firmware of one or many devices (syntax: <code>id=&lt;serial&gt;</code>, <code>&lt;id&gt;</code> or <code>id=&lt;product&gt;</code>, <code>&lt;id&gt;</code>).</td>
</tr>
<tr>
<td><code>set_processor_idle_states</code> or <code>spis</code></td>
<td>Changes the C1, C2 and C3 states for ALL processors in the current system (syntax: <code>spis=1</code> or <code>spis=0</code>).</td>
</tr>
<tr>
<td><code>set_userset_persistence</code> or <code>sup</code></td>
<td>Sets the persistency of UserSet settings during firmware updates (syntax: <code>sup=1</code> or <code>sup=0</code>).</td>
</tr>
<tr>
<td><code>update_fw</code> or <code>ufw</code></td>
<td>Updates the firmware of one or many devices.</td>
</tr>
<tr>
<td><code>update_fw_file</code> or <code>ufwf</code></td>
<td>Updates the firmware of one or many devices. Pass a full path to a text file that contains a serial number or a product type per line.</td>
</tr>
<tr>
<td><code>custom_genicam_file</code> or <code>cgf</code></td>
<td>Specifies a custom GenICam file to be used to open devices for firmware updates. This can be useful when the actual XML on the device is damaged/invalid.</td>
</tr>
<tr>
<td><code>update_kd</code> or <code>ukd</code></td>
<td>Updates the kernel driver of one or many devices.</td>
</tr>
<tr>
<td><code>ipv4_mask</code></td>
<td>Specifies an IPv4 address mask to use as a filter for the selected update operations. Multiple masks can be passed here separated by semicolons.</td>
</tr>
<tr>
<td><code>fw_file</code></td>
<td>Specifies a custom name for the firmware file to use.</td>
</tr>
<tr>
<td><code>fw_path</code></td>
<td>Specifies a custom path for the firmware files.</td>
</tr>
<tr>
<td><code>log_file</code> or <code>lf</code></td>
<td>Specifies a log file storing the content of this text control upon application shutdown.</td>
</tr>
<tr>
<td><code>quit</code> or <code>q</code></td>
<td>Ends the application automatically after all updates have been applied.</td>
</tr>
<tr>
<td><code>force</code> or <code>f</code></td>
<td>Forces a firmware update in unattended mode, even if it isn’t a newer version.</td>
</tr>
<tr>
<td>*</td>
<td>Can be used as a wildcard, devices will be searched by serial number AND by product. The application will first try to locate a device with a serial number matching the specified string and then (if no suitable device is found) a device with a matching product string.</td>
</tr>
</tbody>
</table>
The number of commands that can be passed to the application is not limited.

9.2.6.1 Sample (Windows)

\begin{Verbatim}
\texttt{mvDeviceConfigure ufw=BF000666}
\end{Verbatim}

This will update the firmware of a mvBlueFOX with the serial number BF000666.

\begin{Verbatim}
\texttt{mvDeviceConfigure update_fw=BF*}
\end{Verbatim}

This will update the firmware of ALL mvBlueFOX devices in the current system.

\begin{Verbatim}
\texttt{mvDeviceConfigure update_fw=mvBlueFOX-2* if=output.txt quit}
\end{Verbatim}

This will update the firmware of ALL mvBlueFOX-2 devices in the current system, then will store a log file of the executed operations and afterwards will terminate the application.

\begin{Verbatim}
\texttt{mvDeviceConfigure setid=BF000666.5}
\end{Verbatim}

This will assign the device ID '5' to a mvBlueFOX with the serial number BF000666.

\begin{Verbatim}
\texttt{mvDeviceConfigure ufw=*}
\end{Verbatim}

This will update the firmware of every device in the system.

\begin{Verbatim}
\texttt{mvDeviceConfigure ufw=BF000666 ufw=BF000667}
\end{Verbatim}

This will update the firmware of 2 mvBlueFOX cameras.

\begin{Verbatim}
\texttt{mvDeviceConfigure ipv4_mask=169.254.*;192.168.100* update_fw=GX*}
\end{Verbatim}

This will update the firmware of all mvBlueCOUGAR-X devices with a valid IPv4 address that starts with '169.254.' or '192.168.100.'.
10.1 Introduction

The Hardware Real-Time Controller (HRTC) is built into the FPGA. The user can define a sequence of operating steps to control the way how and when images are exposed and transmitted. Instead using an external PLC, the time critical acquisition control is directly built into the frame grabber. This is a very unique and powerful feature.

10.1.1 Operating codes

The operating codes for each step can be one of the followings:

<table>
<thead>
<tr>
<th>OpCode</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nop</td>
<td>-</td>
<td>No operation</td>
</tr>
<tr>
<td>SetDigout</td>
<td>Operation array on dig out</td>
<td>Set a digital output</td>
</tr>
<tr>
<td>WaitDigin</td>
<td>State definition array on dig in</td>
<td>Wait for a digital input</td>
</tr>
<tr>
<td>WaitClocks</td>
<td>Time in us</td>
<td>Wait a defined time</td>
</tr>
<tr>
<td>Jump</td>
<td>HRTC program address</td>
<td>Jump to any step of the program</td>
</tr>
<tr>
<td>TriggerSet</td>
<td>Frame ID</td>
<td>Set internal trigger signal to sensor controller</td>
</tr>
<tr>
<td>TriggerReset</td>
<td>-</td>
<td>Reset internal trigger signal to sensor controller</td>
</tr>
<tr>
<td>ExposeSet</td>
<td>-</td>
<td>Set internal expose signal to sensor controller</td>
</tr>
<tr>
<td>ExposeReset</td>
<td>-</td>
<td>Reset internal expose signal to sensor controller</td>
</tr>
<tr>
<td>FrameNrReset</td>
<td>-</td>
<td>Reset internal sensor frame counter</td>
</tr>
</tbody>
</table>

256 HRTC steps are possible.

10.1.2 Program controls

The Hardware Real-Time Controller also supports a basic program control mechanism as the following sample shows:

0. \( a = 10 \)
1. \( a = a - 1 \)
2. WaitClocks 50
3. Jump 0 While \( a > 0 \)

The section How to use the HRTC (p. 106) should give the impression what everything can be done with the HRTC.

wxPropView - Introduction:  

MATRIX VISION GmbH
10.2 How to use the HRTC

To use the HRTC you have to set the trigger mode and the trigger source. With object orientated programming languages the corresponding camera would look like this (C++ syntax):

```cpp
CameraSettingsFrameGrabber->triggerMode.writeS("On");
CameraSettingsFrameGrabber->triggerSource.writeS("HRTCtrl_0");
CameraSettingsFrameGrabber->triggerActivation.writeS("dtmOnFallingEdge");
```

When working with wxPropView [p.48] this are the properties to modify in order to activate the evaluation of the HRTC program:

![Figure 1: wxPropView - Setting up the HRTC usage](image)

Following trigger modes can be used with HRTC:

- OnLowLevel
- OnHighLevel
- OnFallingEdge
- OnRisingEdge
- OnHighExposure

Further details about the mode are described in the API documentation:
11 C developers

The description for the mvIMPACT Acquire SDK for C developers is available as a separate file: mvIMPACT Acquire_API_C_manual.chm which is

- either part of the installed package or

Here an online version of the documentation is available as well.
12 C++ developers

The description for the mvIMPACT Acquire SDK for C++ developers is available as a separate file: `mvIMPACT_Acquire_API_CPP_manual.chm` which is

- either part of the installed package or

Here an online version of the documentation is available as well.
The description for the mvIMPACT Acquire SDK for .NET developers is available as a separate file: mvIMPACT\_Acquire_API_NET_manual.chm which is

- either part of the installed package or

Here an online version of the documentation is available as well.
14 Python developers

Note

There is no separate manual available for the Python API right now. For documentation please refer to the C++ manual instead. Almost everything stated there will be valid for Python as well!

14.1 Introduction

As supporting all the various distributions and versions of Python out there with a binary interface is almost impossible the mvIMPACT acquire Python API needs to be compiled for a specific version of Python it shall be used with. Because of that what is shipped at the moment is source code that has been generated using SWIG which before using it requires a compilation operation.

This requires a compiler matching the version of Python that shall be used. E.g. to use mvIMPACT Acquire with Python 2.7 on Windows systems requires Visual Studio 2008. Additional information about the compiler that works for a particular Python version on a particular platform can be found online. See e.g.

- https://docs.python.org/2/extending/building.html
- https://wiki.python.org/moin/WindowsCompilers

Note

Microsoft provides free-of-charge a ‘Microsoft Visual C++ Compiler for Python 2.7’ which can be used to compile ‘mvIMPACT Acquire for python’ for Python versions 2.7 to 3.2:
The use of this compiler packet has been tested by MATRIX VISION and it is is highly recommended.

When installing the mvIMPACT Acquire Python API on a target system all files needed for building the actual extension module can be found in /mvIMPACT_Python.

14.2 Building

During the compilation process Python’s distutils package will be used

14.2.1 Windows

On Windows systems running /mvIMPACT_Python/compileWrapperCode.bat will build and install the Python API in the site-packages sub-folder of your Python installation provided a matching compiler could be found.

Note

The script will assume the Python interpreter can be found in the systems Path variable. If this is not the case you need to append the path to the directory containing Python.exe to this variable either permanently using the Systems environment variable dialog or temporary within the command shell you are calling the script from like this:

```
set Path=%Path%;C:\Python27
compileWrapperCode.bat
```

The command shell you are calling the compilation script from depending on the version of Windows you are working with and the folder you have installed the mvIMPACT Acquire package to might require elevated rights thus you might need to start the command shell with the Run as administrator option.
14.3 Using

14.2.2 Linux

On Linux systems running `mvIMPACT_Python/setup.py` can be used to build and install the Python API in the `site-packages` sub-folder of your Python installation providing a matching compiler could be found. However one must be familiar with Python’s `distutils` package.

**Note**

It is mandatory that the `python-dev` package is installed on the target Linux system, otherwise the binaries cannot be built!
The invoking user of the script must have the rights to install the generated binaries in the python directory of his system. If this is not the case, a recommended way to call the installation script is:

```
sudo -E python setup.py install
```

The building process may take literally a few minutes, so please be patient!

**Attention**

The SWIG generated wrapper code source file is very large! On some embedded systems or on 32-bit systems gcc might have trouble to digest this file as it is known to consume a lot of memory per source line in a given translation unit. When you encounter such a situation please get in touch with us!

14.3 Using

The actual API is almost the same as in C++ thus for now the C++ manual can be used as a reference and function description. There are just some minor differences between the C++ and the Python API which shall be explained here briefly:

- Stuff that has been declared deprecated at the time of publishing the Python API will not be available
- Simple `getter` functions may be wrapped as Python properties to have a more Python-like interface. So e.g. the function `Component::isValid()` will be a property in Python
- Code that in C++ resides in sub-namespaces like e.g. `mvIMPACT::acquire::GenICam` will all end up in `acquire` in Python (this is likely to change in future versions!)
- Some functions that use Python style built-in names like `mvIMPACT::acquire::Component::type()` will use a slightly different name in Python like `getType` in order to avoid confusion

Apart from that if someone is familiar with the C++ interface it shouldn’t be too difficult to use the API. This is how an acquisition from a user selectable device can be done:

```python
from __future__ import print_function
import os
import platform
import string
import sys
# import all the stuff from mvIMPACT Acquire into the current scope
from mvIMPACT import acquire
# import all the mvIMPACT Acquire related helper function such as ‘conditionalSetProperty’ into the current scope
# If you want to use this module in your code feel free to do so but make sure the ‘Common’ folder resides in
from Common import *

# For systems with NO mvDisplay library support
# import ctypes
# import Image
```
# import numpy

devMgr = acquire.DeviceManager()
for i in range(devMgr.deviceCount()):
    pDev = devMgr.getDevice(i)
    print('
' + str(i) + ']: ' + pDev.serial.read() + '(', pDev.product.read(), pDev.family.read(), end='')
    if pDev.interfaceLayout.isValid:
        conditionalSetProperty(pDev.interfaceLayout, acquire.dilGenICam)
        print(', interface layout: ' + pDev.interfaceLayout.readS(), end='')
    if pDev.acquisitionStartStopBehaviour.isValid:
        conditionalSetProperty(pDev.acquisitionStartStopBehaviour, acquire.assbUser)
        print(', acquisition start/stop behaviour: ' + pDev.acquisitionStartStopBehaviour.readS(), end='')
    if pDev.isInUse():
        print(', !!!ALREADY IN USE!!!', end='')
    print(')')

print('Please enter the number in front of the listed device followed by [ENTER] to open it: ', end='')

devNr = int(raw_input())
if (devNr < 0) or (devNr >= devMgr.deviceCount()):
    print('Invalid selection!')
sys.exit(-1)

pDev = devMgr.getDevice(devNr)
pDev.open()

print('Please enter the number of buffers to capture followed by [ENTER]: ', end='')
framesToCapture = int(raw_input())
if framesToCapture < 1:
    print('Invalid input! Please capture at least one image')
sys.exit(-1)

# The mvDisplay library is only available on Windows systems for now
isDisplayModuleAvailable = platform.system() == "Windows"
if isDisplayModuleAvailable:
    display = acquire.ImageDisplayWindow("A window created from Python")
else:
    print("The mvIMPACT Acquire display library is not available on this(' + platform.system() + ') system. Consider using... and numpy(Numerical Python) packages instead. Have a look at the source code of this application to get an idea how.")

# For systems with NO mvDisplay library support
channelType = numpy.uint16 if channelBitDepth > 8 else numpy.uint8
fi = acquire.FunctionInterface(pDev)
statistics = acquire.Statistics(pDev)

while fi.imageRequestSingle() == acquire.DMR_NO_ERROR:
    print("Buffer queued")
    pPreviousRequest = None

    manuallyStartAcquisitionIfNeeded(pDev, fi)
    for i in range(framesToCapture):
        requestNr = fi.imageRequestWaitFor(-1)
        if fi.isRequestNrValid(requestNr):
            pRequest = fi.getRequest(requestNr)
            if pRequest.isValid():
                if isDisplayModuleAvailable:
                    display.GetImageDisplay().SetImage(pRequest)
                    display.GetImageDisplay().Update()
                else:
                    cbuf = (ctypes.c_char * imageSize).from_address(long(req.imageData.read()))
                    arr = numpy.fromstring(cbuf, dtype = channelType)
                    if channelCount == 1:
                        if isDisplayModuleAvailable:
                            image = Image.frombuffer('1', (height, width), pRequest.data, 'raw', channelType, 0)
                        else:
                            image = Image.frombuffer('1', (height, width), channelType, 'raw', channelType, 0)
                    else:
                        if isDisplayModuleAvailable:
                            image = Image.frombuffer('RGBA' if alpha else 'RGB', (height, width), pPreviousRequest.data, 'raw', channelType, 0)
                        else:
                            image = Image.frombuffer('RGBA' if alpha else 'RGB', (height, width), channelType, 'raw', channelType, 0)
pPreviousRequest = pRequest
fi.imageRequestSingle()
else:
    print("imageRequestWaitFor failed (" + str(requestNr) + ", " + ImpactAcquireException.getErrorCodeAsString(requestNr) + ")")
manuallyStopAcquisitionIfNecessary(pDev, fi)
raw_input("Press Enter to continue...")

**Note**

The above code uses the Python 3 style `print`. Because of the line

```python
from __future__ import print_function
```

This will also work with Python versions starting with version 2.6. For smaller versions of Python the code needs to be changed!
15 DirectShow Interface

Note

DirectShow can only be used in combination with the Microsoft Windows operating system. Since Windows Vista, Movie Maker does not support capturing from a device registered for DirectShow anymore.

This is the documentation of the MATRIX VISION DirectShow_acquire interface. A MATRIX VISION specific property interface based on the IKsPropertySet has been added. All other features are related to standard DirectShow programming.

- Supported Interfaces (p.114)
- Logging (p.114)
- Registering and renaming devices for DirectShow usage (p.115)

15.1 Supported Interfaces

15.1.1 IAMCameraControl

15.1.2 IAMDroppedFrames

15.1.3 IAMStreamConfig

15.1.4 IAMVideoProcAmp

15.1.5 IKsPropertySet

The DirectShow_acquire supports the IKsPropertySet Interface. For further information please refer to the Microsoft DirectX 9.0 Programmer's Reference.

Supported property set GUID's:

- AMPROPERTY_PIN_CATEGORY
- DIRECT_SHOW_ACQUIRE_PROPERTYSET

15.1.6 ISpecifyPropertyPages

15.2 Logging

The DirectShow_acquire logging procedure is equal to the logging of the MATRIX VISION products which uses mvIMPACT Acquire. The log output itself is based on XML.

If you want more information about the logging please have a look at the Logging chapter of the respective "mvIMPACT Acquire API" manual.
15.3 Registering and renaming devices for DirectShow usage

Note

Please be sure to register the MV device for DirectShow with the right version of mvDeviceConfigure (p. 95). I.e. if you have installed the 32 bit version of the VLC Media Player, Virtual Dub, etc., you have to register the MV device with the 32 bit version of mvDeviceConfigure (p. 95) ("C:\Program Files\MATRIX VISION\mvIM←PACT Acquire\bin")!

15.3.1 Registering devices

To register a device/devices for access under DirectShow please perform the following registration procedure:

1. Start mvDeviceConfigure.
   If no device has been registered the application will more or less (depending on the installed devices) look like this.

   ![Figure 1: mvDeviceConfigure - start window](image)

   - Right click on a device entry to get a menu with available options for the selected device.
   - An entry is disabled if the underlying feature is not available for the selected device.
   - Double click on a list entry to start live acquisition from this device in a new instance of mvPropView (mvPropView must be locatable via the systems path or must reside in the same directory as mvDeviceConfigure).
   - Menu entries under ‘Roben’ will be enabled and disabled whenever the currently selected device changes.
   - To modify the way log outputs are created select “Action → Configure Log Output”.

2. To register every installed device for DirectShow access click on the menu item "DirectShow" → "Register all devices".
3. After a successful registration the column "registered for DirectShow" will display 'yes' for every device and the devices will be registered with a default DirectShow friendly name.
15.3.2 Renaming devices

If you want to modify the friendly name of a device under DirectShow, please perform the following procedure:

1. If mvDeviceConfigure is already not running, please start it.
2. Now, select the device you want to rename, click the right mouse button and select "Set DirectShow friendly name":

3. Then, a dialog will appear. Please enter the new name and confirm it with "OK".
4. Afterwards the column "DirectShow friendly name" will display the newly assigned friendly name.

**Note**

Please do not select the same friendly name for two different devices. In theory this is possible, however the mvDeviceConfigure GUI will not allow this to avoid confusion.

### 15.3.3 Make silent registration

To make a silent registration without dialogs, the Windows tool "regsvr32" via command line can be used.

The following command line options are available and can be passed during the silent registration:

**EXAMPLES:**

Register ALL devices that are recognized by mvIMPACT Acquire (this will only register devices which have drivers installed).

```
regsvr32 <path>\DirectShow_acquire.ax /s
```
## 16  Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A/D reference</td>
<td>Upper threshold of video signal to be digitized. All values above this limit value are digitized to 255. Increasing the reference level results in contrast deterioration and vice versa.</td>
</tr>
<tr>
<td>ADC</td>
<td>Analog to digital converter (A/D converter)</td>
</tr>
<tr>
<td>Resolution</td>
<td>Number of pixels (horizontal x vertical)</td>
</tr>
<tr>
<td>Base address</td>
<td>Starting address from which the memory or register are inserted.</td>
</tr>
<tr>
<td>Image refresh rate</td>
<td>Number of transferred images per second. Normally specified in Hz (e.g. 70 Hz)</td>
</tr>
<tr>
<td>Bpp</td>
<td>Bits per pixel</td>
</tr>
<tr>
<td>Bus</td>
<td>A group line via which the various parts of the computer communicate with one another.</td>
</tr>
<tr>
<td>CCIR</td>
<td>Comité Consultatif International of the Radio Communications European video standard for 50 Hz gray scale.</td>
</tr>
<tr>
<td>Clamp signal</td>
<td>Clamp signal means, that a AC coupled video signal is clamped on the porch to get a signal transfer with less noise and independent from the d.c. voltage portion.</td>
</tr>
<tr>
<td>CPU</td>
<td>Central processing unit</td>
</tr>
<tr>
<td>DAC</td>
<td>Digital to analog converter (D/A converter)</td>
</tr>
<tr>
<td>Defaults</td>
<td>Standard system settings</td>
</tr>
<tr>
<td>DIO</td>
<td>Digital inputs and outputs</td>
</tr>
<tr>
<td>DIP switch</td>
<td>Dual inline package (housing design)</td>
</tr>
<tr>
<td>External trigger</td>
<td>External event used to initiate image capture.</td>
</tr>
<tr>
<td>False colors</td>
<td>Colors are assigned to gray scale via a look-up table. This allows even small gray scale differences can be displayed clearly.</td>
</tr>
<tr>
<td>Field</td>
<td>All odd lines of a field (odd field) or all even lines of a field (even field) of an interlaced video image.</td>
</tr>
<tr>
<td>Frame grabber</td>
<td>Here: PC plug-in card for digitization and storage of video images.</td>
</tr>
<tr>
<td>Host</td>
<td>Here: the PC</td>
</tr>
<tr>
<td>Interlaced</td>
<td>Interlacing method; conforming to the television standard, this method involves acquiring two fields in succession (all odd lines, all even lines) and combining them to create a frame. The result is greatly reduced flicker during on-screen display.</td>
</tr>
<tr>
<td>Interrupt</td>
<td>Interrupt signal sent to the processor. The program currently running is interrupted and a predefined function is executed.</td>
</tr>
<tr>
<td>ISR</td>
<td>Interrupt service routine</td>
</tr>
<tr>
<td>IRQ</td>
<td>Interrupt request</td>
</tr>
<tr>
<td>Look-up table</td>
<td>Table of assignments. Here, new gray scale or colors are normally assigned to gray scale. Look-up tables can, however, also be used for any other math functions.</td>
</tr>
<tr>
<td>LSB</td>
<td>Least significant bit</td>
</tr>
<tr>
<td>LUT</td>
<td>Look-up table</td>
</tr>
<tr>
<td>Monochrome</td>
<td>A single-color (black and white) image</td>
</tr>
<tr>
<td>MSB</td>
<td>Most significant bit</td>
</tr>
<tr>
<td>Non-interlaced</td>
<td>Image acquisition and output line by line</td>
</tr>
<tr>
<td>NTSC</td>
<td>National Television Standard Code. US video standard for 60 Hz colors.</td>
</tr>
<tr>
<td>Overlay</td>
<td>Image memory for outputting text and graphics via the video monitor.</td>
</tr>
<tr>
<td>PAL</td>
<td>Phase alteration line; 50 Hz video standard for color.</td>
</tr>
<tr>
<td>Pixels</td>
<td>Picture element</td>
</tr>
<tr>
<td>PoCL</td>
<td>Power over CameraLink - The cameras are powered over CL cable and therefore need no additional power supply. The mvHYPERIONs which supports PoCL, are &quot;Switchable PoCL frame grabbers&quot; as described in the CameraLink™ specification. This means that both camera and cable have to support PoCL otherwise Pin 1 and Pin 26 of the CL connectors act like internal shields.</td>
</tr>
<tr>
<td>Pseudo colors</td>
<td>Display of gray scale images in false colors. A corresponding color is assigned to a specific gray scale value.</td>
</tr>
<tr>
<td><strong>Square pixels</strong></td>
<td>Square-shaped pixels (height-width ratio 1:1)</td>
</tr>
<tr>
<td><strong>RS170</strong></td>
<td>US video standard for 60 Hz b/w colors</td>
</tr>
<tr>
<td><strong>TFT display</strong></td>
<td>Thin film transistor display</td>
</tr>
<tr>
<td><strong>True color</strong></td>
<td>24-bit true color; 16.7 million colors</td>
</tr>
<tr>
<td><strong>Vertical sync</strong></td>
<td>Synchronization pulse in video signal for field end recognition.</td>
</tr>
<tr>
<td><strong>Zero signal</strong></td>
<td>The zero signal was needed with the old frame grabbers, to calibrate the analog/digital converter (ADC) (signal and parameter aren't important anymore).</td>
</tr>
</tbody>
</table>
17 Use cases

17.1 scanCameras Working with line scan cameras

There are several use cases concerning line scan camera:

- "Camera synchronization issues" :
  - Pass-through of digital input signals (p. 121)
  - Working with pulse start events (p. 122)
  - Working with an rotary encoder (p. 123)
  - Working with a Basler Sprint line scan color camera (p. 126)

- "Trigger issues" :
  - Working with trigger events (p. 134)
  - Synchronous acquisition with different camera settings (p. 139)

17.2 Pass-through of digital input signals

The mvHYPERION offers input and output signals which do not have anything in common electrically. In some cases it could be necessary to link them electrically, e.g. CC1 with Sync-In (J3).

If someone puts e.g. a periodical digital signal at the Sync-In input, this signal will be bypassed to CC1. The following Figure shows how this will look like using wxPropView (p. 48):

![Figure 1: wxPropView - Digital I/O - Pass-through of a digital signal](image-url)
17.3 Working with pulse start events

The pulse start configuration list represents a Hardware Real-Time Controller (HRTC) as known from USB 2.0 camera mvBlueFOX. The mvHYPERION features two HRTC's (PulseStartConfiguration0 and PulseStartConfiguration1). A HRTC generates signals at the digital outputs, which can be defined by the user.

Currently you can use the HRTC in three ways:

- **Periodical repeat of the pulse sequence**
  - parameters:
    - frequency of the pulse sequence [Hz]

- **Single run through**
  - parameters:
    - digital input or on-board signal, which starts the pulse sequence initially -> "DigitalSignal"
    - falling or rising edge of the digital input -> "TriggerMoment"

The following sample shows, how a SinglePulse with 41 us width at CC1 is defined, which will start after a FallingEdge at the Sync-In input:

![Figure 1: wxPropView - Pulse start sample](image)

- **Working with an rotary encoder** (p. 123)
  - parameters:
    - external signals from a rotary encoder, which starts the pulse sequence initially -> "RotaryDecoder"
    - decode the signals of the rotary encoder.
17.4 Working with an rotary encoder

In many applications including industrial controls, robotics etc. an rotary encoder (or incremental encoder) is used, which is an electro-mechanical device that converts the angular position of a shaft or axle to an analog or digital code, making it an angle transducer.

During the rotation of the axis, an rotary encoder generates a so-called Gray code signal at the output of the two data lines:

**Figure 1: Principle of an rotary encoder**

**Figure 2: Gray code signal at the output of the two data lines**
As described in Working with pulse start events (p.122) the mvHYPERION features two HRTC’s (PulseStartConfiguration0 and PulseStartConfiguration1), which also features a Rotary Decoder (p.123).

The Rotary Decoder offers some parameters:

- DigitalSignalA (p.124)
- DigitalSignalB (p.124)
- PulseMultiplication (p.125)
- Direction (p.125)
- Mode (p.125)
- int Reset() (p.126)

17.4.1 DigitalSignalA

The first data line of the rotary encoder.

17.4.2 DigitalSignalB

The second data line of the rotary encoder.
17.4 Working with an rotary encoder

17.4.3 Pulse Multiplication

It is possible to multiply the pulse sequence. The difference is shown in the following figure:

Figure 4: Pulse multiplication

17.4.4 Direction

Specifies the direction of rotation.

17.4.5 Mode

Here you can specify the mode of the encoder. There are three different modes:

- "NoInhibit": The direction of rotation is not relevant.
- "InhibitBackward": The direction of rotation is relevant. This means, if the direction is changed, the edges will be ignored.
- "InhibitBackwardUntilLastPos": The direction of rotation is relevant and the edge changes are counted. This means, if the direction is changed, the edges will be counted (counter initial value is 0). If the direction changes again, the counter will be decremented until 0.
Note

With the property "Divider" you can set if every signal (or every second, third etc.) from the rotary encoder is used.

17.4.6 int Reset()

With this function you can reset the counter, which is saved in InhibitBackwardUntilLastPos (p. 125).

17.5 Working with a Basler Sprint line scan color camera

17.5.1 Introduction

Basler’s Sprint line scan color camera uses a two line sensor with a Bayer Mosaic filter mask. To use the demosaic algorithm correctly, you have to set the

- BayerParity.

There are 4 possibilities for the BayerParity:
• "Red-green"
• "Green-red"
• "Blue-green"
• "Green-blue"

To guarantee the BayerParity start condition, the camera sends a FrameValid synchronization signal. Thus, you can say that the camera behaves like a area-scan camera.

With Basler’s camera control tool you can define the frame height (always a multiple of 2) and therefore the FrameValid signal.

In combination with a triggered image acquisition, you have to keep in mind that an image acquisition will start after

• a start trigger signal (e.g. a light barrier) and
• a valid FrameValid signal.

The mvHYPERION frame grabbers are suitable for these applications.

![Figure 1: Triggered acquisition with area-scan cameras](image)

Note

As you can see in Figure 1, there are trigger delays possible (up to an almost complete frame).

The trigger start condition (TriggerActivation) can be set via wxPropView (p. 48) in "Setting -> Base -> Camera -> TriggerControls". In Figure 2, the start condition is a RisingEdge at the J4 connector.
The following sections show how you have to set the mvHYPERION according to the different modes.

17.5.2 RawLineAcquisition Mode

In this mode, the camera sends the color lines alternately. For more details about this mode, please have a look at the camera's manual.
Although we defined a line scan camera in the ScanMode, the mvHYPERION has to handle the FrameValid synchronization. With line scanStartCondition the mvHYPERION runs in a mode, which waits for a FrameValid signal when the image starts. All further FrameSync signals are ignored. This mode is required with Basler’s sprint cameras.

The camera line synchronization in this sample is a FallingEdge at Sync-In. Afterwards using PulseStart←Configuration0, a FallingEdge SinglePulse signal is generated at digital out CC1:
17.5.3 EnhancedRawLineAcquisition Mode

The camera also offers a special mode called "Enhanced Raw Line Acquisition Mode". It provides a raw "green" pixel value for each point of an imaged object and, in addition, either a raw "red" or a raw "blue" pixel value (for more details please have a look at the camera's manual). In sum, you can say that this is a double exposure of the same line. For this, you will need a pulse multiplication.
17.5 Working with a Basler Sprint line scan color camera

It only makes sense to reproduce this application using a rotary encoder (p. 123). The application could look like this:

![wxPropView - Settings Height (H), Format BayerPacked, BayerParity, ScanMode, line scanStartCondition](image_url)

---

It only makes sense to reproduce this application using a **rotary encoder** (p. 123). The application could look like this:
The camera line synchronization in this sample is the trigger signal of a rotary encoder. Afterwards, two *SinglePulse* signals are sent via digital out *CC1*.
17.5 Working with a Basler Sprint line scan color camera

Figure 7: wxPropView - Triggered acquisition with area-scan cameras

<table>
<thead>
<tr>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>With the property &quot;Divider&quot; you can set if every signal (or every second, third etc.) from the rotary encoder is used.</td>
</tr>
</tbody>
</table>
17.6 Working with trigger events

One or many trigger(s) can be used to control start of an acquisition, of a frame of an acquisition or each line of a frame (for line scan devices). It can also be used to control the exposure duration at the beginning of a frame.

Typically, a trigger event has a

- falling edge and a
- rising edge.

A falling edge initializes the capture of a frame (see Figure 1).

![Figure 1: Typical trigger events](image)

There are three modes to use trigger events with mvHYPERION frame grabbers:

- **FrameStart** (p.135)
- **FrameStart + FrameStop** (p.136)
- **FrameStart + FrameStop (with restart option using the same event)** (p.137)

The three modes will be described on the basis of a trigger event on J3.

Please open "Setting -> Base -> Camera -> TriggerControls", where you can manage the mentioned trigger events.
17.6 Working with trigger events

17.6.1 FrameStart

In this mode, a frame is captured after a trigger event, which can be specified by the user in "TriggerActivation". Figure 2 shows a falling edge. The "TriggerMode" has to be "On" and the correct "TriggerSource" has to be specified (in our example J3). "TriggerMode" of FrameStop has to be "Off".

While the frame is not finished, other trigger events are ignored.

![Figure 2: Trigger example](image)

It is also possible to define a trigger delay via

- "TriggerDelayAbs_us" or
- "TriggerDelayLines", which only makes sense using line scan cameras.

![Figure 3: wxPropView - Possible trigger sources](image)
136 CONTENTS

With the property "TriggerDivider" it is possible to define, which trigger events are regarded (see Figure 5).

17.6.2 FrameStart + FrameStop

In this mode you have two signal definitions. With the appearance of the stop event, the image acquisition will be stopped line-synchronously and the amount of acquired lines will be available as result or image information.

Figure 7 shows an example definition, where the FrameStart is on a rising edge of J3 and FrameStop is on a falling edge of J4.
17.6 Working with trigger events

### 17.6.2.1 FrameStart + FrameStop (with restart option using the same event)

In this mode every trigger event is accepted at any time during active image acquisition. If a trigger event happens during an acquisition, the acquisition will be stopped line-synchronously even if all requested lines of the previous image have not been acquired completely. The amount of lines will be available as result or image information.

You have to select the same parameters using the same event in "FrameStart" and "FrameStop" to activate this mode.
Using line scan cameras, it is possible to define a delay using "TriggerDelayLines". With it you can shift, for example, the FrameStart.

### 17.6.3 AcquisitionStart

Some cameras like line scan cameras send "FrameValid" or "LineValid" as a signal for "FrameStart". Anyway, it is necessary to have the possibility to set the start of the acquisition. This can be done using "AcquisitionStart".
17.7 Synchronous acquisition with different camera settings

For details, how you can create a synchronous acquisition with different camera settings is described in section Setting up multiple display support and/or work with several capture settings in parallel (p. 51).

It is also possible to control the synchronous acquisition with the frame grabber’s HRTC (p. 106).

For this, you have

1. to set the “TriggerSource” (“Setting”) of both capture settings to e.g. "HRTC0_0" and select the “Trigger Activation” e.g. "FallingEdge". This means that the FrameStart will start after a falling edge from the HRTC.

2. Then, you have to set the "Digital I/O -> Digital Outputs -> ControlMode" to "HardwareRealTimeController".
3. Now, you can create a HRTC program in "Digital I/O -> HardwareRealTimeController".

The HRTC program consist of 5 steps:

(a) Wait for an "Off" signal at the digital input (in this sample at position 9 which is DigIn0; Figure 4).
(b) Wait for an "On" signal at the digital input (in this sample at position 9 which is DigIn0; Figure 4).
(c) Then, a pulse signal is set to "On" at the "TriggerController" (in this sample at position 9 which is shown in the help window).

(d) Afterwards, the pulse signal is "Off" again.
(e) Finally, the program jumps back to step 0.

4. Now, if you have a rotary encoder and you have connected the signal A to the “Sync-In”, you can bypass this signal, for example, to CC1:
Now, if all two capture settings for both cameras have

- the "TriggerSource" set to "HRTCtrl_0",
- the "TriggerMode" to "On" and
- the "TriggerActivation" set to "FallingEdge",

as shown in Figure 2 for the "Base" setting, both cameras will capture an image at the same time.