Contents

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

2 Imprint ........................................ 7

3 Revisions ..................................... 8

4 Graphic Symbols .............................. 12
   4.1 Notes, Warnings, Attentions ........... 12
   4.2 Webcasts ................................ 12

5 Important information ....................... 13
   5.1 High-Speed USB design guidelines ...... 14
   5.2 European Union Declaration of Conformity statement ............. 14
   5.3 Legal notice .............................. 17
       5.3.1 For customers in the U.S.A. ...... 17
       5.3.2 For customers in Canada .......... 18
       5.3.3 Pour utilisateurs au Canada ...... 18

6 Introduction ................................ 19
   6.1 Order code nomenclature ............... 19
       6.1.1 mvBlueFOX ......................... 19
       6.1.2 mvBlueFOX-M ..................... 20
       6.1.3 mvBlueFOX-IGC ................... 20
       6.1.4 mvBlueFOX-MLC ................... 21
   6.2 What’s inside and accessories .......... 22
# Quickstart

## Windows

1. **System Requirements**
2. **Installing the mvIMPACT Acquire driver**
3. **Installing the hardware**

## Linux

1. **System Requirements**
2. **Installing the mvIMPACT Acquire driver**
3. **Installing the hardware**

## Relationship between driver, firmware and FPGA file

1. **FPGA**
2. **Firmware**

## Settings behaviour during startup

## Technical data

### Power supply

### Standard version (mvBlueFOX-xxx)

1. **Dimensions and connectors**
2. **LED states**
3. **Components**

### Board-level version (mvBlueFOX-Mxxx)

1. **Dimensions and connectors**
2. **LED states**
3. **Components**
4. **Accessories mvBlueFOX-Mxxx**

### Single-board version (mvBlueFOX-MLC2xx)

1. **Typical Power consumption @ 5V**
2. **Dimensions and connectors**
3. **LED states**
4. **Assembly variants**

### Single-board version with housing (mvBlueFOX-IGC2xx)

1. **Dimensions and connectors**
2. **LED states**
3. **Positioning tolerances of sensor chip**
16 Python developers

16.1 Introduction .................................................. 130

16.2 Building ....................................................... 130

16.2.1 Windows ................................................... 130

16.2.2 Linux ....................................................... 131

16.3 Using .......................................................... 131

17 DirectShow Interface ........................................... 134

17.1 Supported Interfaces ........................................ 134

17.1.1 IAMCameraControl ...................................... 134

17.1.2 IAMDroppedFrames ..................................... 134

17.1.3 IAMStreamConfig ........................................ 134

17.1.4 IAMVideoProcAmp ...................................... 134

17.1.5 IKsPropertySet ........................................... 134

17.1.6 ISpecifyPropertyPages ................................. 134

17.2 Logging ....................................................... 134

17.3 Registering and renaming devices for DirectShow usage .... 135

17.3.1 Registering devices ..................................... 135

17.3.2 Renaming devices ....................................... 137

17.3.3 Make silent registration ................................. 138

18 Troubleshooting ................................................ 139

18.1 Accessing log files ......................................... 139

18.1.1 Windows .................................................. 139

18.1.2 Linux ....................................................... 139
19 Use cases

19.1 Introducing acquisition / recording possibilities
19.1.1 Generating very long exposure times
19.1.2 Using VLC Media Player

19.2 Improving the acquisition / image quality
19.2.1 Correcting image errors of a sensor
19.2.2 Optimizing the color fidelity of the camera

19.3 Working with triggers
19.3.1 Using external trigger with CMOS sensors

19.4 Working with HDR (High Dynamic Range Control)
19.4.1 Adjusting sensor \*00W
19.4.2 Adjusting sensor \*02d (-1012d)

19.5 Working with LUTs
19.5.1 Introducing LUTs

19.6 Saving data on the device
19.6.1 Creating user data entries

19.7 Working with device features

19.8 Working with several cameras simultaneously
19.8.1 Using 2 mvBlueFOX-MLC cameras in Master-Slave mode
19.8.2 Synchronize the cameras to expose at the same time

19.9 Working with the Hardware Real-Time Controller (HRTC)
19.9.1 Single camera samples (HRTC)
19.9.2 Multiple camera samples (HRTC)
21 Appendix A.5 CMOS specific camera / sensor data

21.1 mvBlueFOX-[Model]200w (0.4 Mpix [752 x 480])

21.1.1 Introduction .................................................. 232
21.1.2 Details of operation ........................................ 232
21.1.3 Measured frame rates ....................................... 233
21.1.4 Sensor Data .................................................. 234
21.1.5 Device Feature And Property List ......................... 235

21.2 mvBlueFOX-[Model]202a (1.3 Mpix [1280 x 1024])

21.2.1 Introduction .................................................. 235
21.2.2 Details of operation ........................................ 235
21.2.3 Sensor Data .................................................. 237
21.2.4 Characteristics .............................................. 237
21.2.5 Device Feature And Property List ......................... 238

21.3 mvBlueFOX-[Model]202b (1.2 Mpix [1280 x 960])

21.3.1 Introduction .................................................. 238
21.3.2 Details of operation ........................................ 238
21.3.3 Measured frame rates ....................................... 240
21.3.4 Sensor Data .................................................. 240
21.3.5 Device Feature And Property List ......................... 241

21.4 mvBlueFOX-[Model]202d (1.2 Mpix [1280 x 960])

21.4.1 Introduction .................................................. 242
21.4.2 Details of operation ........................................ 242
21.4.3 Measured frame rates ....................................... 243
21.4.4 Sensor Data .................................................. 244
21.4.5 Device Feature And Property List ......................... 245

21.5 mvBlueFOX-[Model]205 (5.0 Mpix [2592 x 1944])

21.5.1 Introduction .................................................. 245
21.5.2 Details of operation ........................................ 245
21.5.3 Measured frame rates ....................................... 247
21.5.4 Sensor Data .................................................. 248
21.5.5 Device Feature And Property List ......................... 249
1 About this manual

1.1 Composition of the manual

This manual is based on a modular concept and contains several individual books. That means like in many object-oriented programming languages for each specific functionality you will have your own “class”. Instead of classes, you have books. For example, if you want to know anything about the GUI based applications, then you have to go to the Application Usage book. If you want to know how images are acquired with your device, have a look in the respective programming language chapter.

Here is a short summary about all books that form this manual:

- The manual starts with technical data about the device like sensors (for cameras) or electrical characteristics as well as a quick start chapter.

Afterwards, you will find the different books:

- The installation package comes with a couple of tools offering a graphical user interface (GUI) to control mvIMPACT Acquire compliant devices.
  - wxPropView can be used to capture image data and to change parameters like AOI or gain
  - mvDeviceConfigure can be used to e.g. perform firmware updates, assign a unique ID to a device that is stored in non-volatile memory or to configure to log-message output.
- HRTC - Hardware Real-Time Controller
  - It is possible to define sequences of operating steps to control acquisition or time critical I/O. This FPGA built-in functionality is called Hardware Real-Time Controller (short: HRTC).
- DirectShow developers
  - This is the documentation of the MATRIX VISION DirectShow_acquire interface.
- Use cases
  - This book offers solutions and explanations for standard use cases.

For C, C++, .NET developers, there are separate mvIMPACT Acquire manuals describing the API. Please refer to the Programming section

These documents can either be downloaded from http://www.matrix-vision.com or come as part of an installation package. The manuals e.g. contain information 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 custom client installers packages for Windows and Linux
- the API itself

1.2 How to get started?

This chapter gives you a short overview, how to get started with your device 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 explains how to get started programming own applications.
1.2 How to get started?

1.2.1 Installation

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

• Windows:
  – Check the system requirements (p. 24).
  – Install the software and driver (p. 25).
  – Install the hardware (p. 27).
  – Configure the mvBlueFOX (p. 78)
    • e.g. make a white balance (p. 104) (color sensors).

• Linux:
  – Check the system requirements (p. 31).
  – Install the software and driver (p. 33).
  – Install the hardware (p. 39).
  – Configure the mvBlueFOX (p. 78)
    • e.g. make a white balance (p. 104) (color sensors).

1.2.2 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: http://www.matrix-vision.com.

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.

1.2.2.1 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>
1.2 How to get started?

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 6.0)</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.dll&quot; (comes with the driver package, but the driver package must be installed AFTER installing NeuroCheck 6.1)</td>
</tr>
</tbody>
</table>

1.2.2.2 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.

1.2.2.3 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.
1.2.2.4 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.

1.2.2.5 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. 135).

1.2.2.6 Micro-Manager support

Every mvIMPACT Acquire compliant device can be operated under https://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: https://micro-manager.org/wiki/MatrixVision.

1.2.2.6.1 code

- https://valelab4.ucsf.edu/trac/micromanager/browser/DeviceAdapters/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 Programming

To understand how to control the device and handle image data you will have a good introduction by reading the main pages of the corresponding mvIMPACT Acquire interface reference. Additionally, please have a look at the example programs. Several basic examples are available. For details please refer to

- the C API (p. 127) section
- the C++ API (p. 128) section
- the .NET API (p. 129) section
- the Python API (p. 130) section

depending on the programming language you will use for your application.
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.abis.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>09. November 2018</td>
<td>Added &quot;Hard Disk Recording&quot; in <strong>wxPropView</strong> (p.78).</td>
</tr>
<tr>
<td>21. December 2016</td>
<td>Updated <strong>Setting up multiple display support and/or work with several capture settings in parallel</strong> (p.95).</td>
</tr>
<tr>
<td>15. December 2016</td>
<td>Added Micro-Manger in <strong>Driver concept</strong> (p.3).</td>
</tr>
<tr>
<td>23. August 2016</td>
<td>Added measured frame rates of sensors <strong>mvBlueFOX-[Model]200w</strong> (0.4 Mpix [752 x 480]) (p.232) <strong>mvBlueFOX-[Model]202b</strong> (1.2 Mpix [1280 x 960]) (p.238) <strong>mvBlueFOX-[Model]202d</strong> (1.2 Mpix [1280 x 960]) (p.242) <strong>mvBlueFOX-[Model]205</strong> (5.0 Mpix [2592 x 1944]) (p.245).</td>
</tr>
<tr>
<td>01. August 2016</td>
<td>Extended use case <strong>Take two images with different expose times after an external trigger (HRTC)</strong> (p.186).</td>
</tr>
<tr>
<td>11. May 2016</td>
<td>Added <strong>Quick Setup Wizard</strong> (p.78).</td>
</tr>
<tr>
<td>02. December 2015</td>
<td>Updated <strong>CE declarations</strong> (p.14).</td>
</tr>
<tr>
<td>27. October 2015</td>
<td>Added <strong>Command-line options</strong> (p.116).</td>
</tr>
<tr>
<td>04. August 2015</td>
<td>Added Windows 10 support.</td>
</tr>
<tr>
<td>23. April 2015</td>
<td>Added use case <strong>Edge controlled triggering (HRTC)</strong> (p.188).</td>
</tr>
<tr>
<td>16. April 2015</td>
<td>Updated supported Windows versions.</td>
</tr>
<tr>
<td>15. April 2015</td>
<td>Added lens protrusion.</td>
</tr>
<tr>
<td>11. March 2015</td>
<td>Added chapter <strong>Accessing log files</strong> (p.103).</td>
</tr>
<tr>
<td>26. February 2015</td>
<td>Moved <strong>Creating double acquisitions (HRTC)</strong> (p.185) to HRTC Use Cases.</td>
</tr>
<tr>
<td>09. January 2015</td>
<td>Extended sample <strong>Using 2 mvBlueFOX-MLC cameras in Master-Slave mode</strong> (p.175).</td>
</tr>
<tr>
<td>10. December 2014</td>
<td>Corrected <strong>Order code nomenclature</strong> (p.19) : mvBlueFOX cameras without filter have the order code 9 (excluding mvBlueFOX-MLC).</td>
</tr>
<tr>
<td>01. December 2014</td>
<td>Extended use case <strong>Using 2 mvBlueFOX-MLC cameras in Master-Slave mode</strong> (p.175).</td>
</tr>
<tr>
<td>25. November 2014</td>
<td>Corrected the possible <strong>HRTC - Hardware Real-Time Controller</strong> (p.125) steps to 256.</td>
</tr>
<tr>
<td>21. October 2014</td>
<td>Added description about the record mode in <strong>How to see the first image</strong> (p.83).</td>
</tr>
<tr>
<td>17. July 2014</td>
<td>Added use case <strong>Introducing LUTs</strong> (p.170).</td>
</tr>
<tr>
<td>25. April 2014</td>
<td>Added description about <strong>Working with the hardware Look-Up-Table (LUT)</strong> (p.115).</td>
</tr>
<tr>
<td>25. March 2014</td>
<td>Added use case <strong>Correcting image errors of a sensor</strong> (p.144).</td>
</tr>
<tr>
<td>18. February 2014</td>
<td>Updated <strong>Characteristics</strong> (p.244) of <strong>mvBlueFOX-[Model]202d</strong> (1.2 Mpix [1280 x 960]) (p.242).</td>
</tr>
<tr>
<td>12. December 2013</td>
<td>Changed figure in <strong>Using 2 mvBlueFOX-MLC cameras in Master-Slave mode</strong> (p.175).</td>
</tr>
<tr>
<td>06. December 2013</td>
<td>Added information about <strong>Changing the view of the property grid to assist writing code that shall locate driver features</strong> (p.102).</td>
</tr>
<tr>
<td>22. November 2013</td>
<td>Extended information in <strong>Adjusting sensor -x00w</strong> (p.163) and <strong>Adjusting sensor -x02d</strong> (-1012d) (p.167).</td>
</tr>
<tr>
<td>30. October 2013</td>
<td>Enhanced cable description in <strong>12-pin Wire-to-Board header (USB 2.0 / Dig I/O)</strong> (p.61).</td>
</tr>
<tr>
<td>Date</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>09. October 2013</td>
<td>Added information about Positioning tolerances of sensor chip (p. 70).</td>
</tr>
<tr>
<td>02. September 2013</td>
<td>Updated Order code nomenclature (p. 19).</td>
</tr>
<tr>
<td>22. April 2013</td>
<td>Added chapter Sensor’s optical midpoint and orientation (p. 61) and corrected feature table in CMOS sensors (p. 73) (software trigger).</td>
</tr>
<tr>
<td>19. March 2013</td>
<td>Updated Figure 4 in chapter Dimensions and connectors (p. 46) and added Figure 5.</td>
</tr>
<tr>
<td>07. December 2012</td>
<td>All parts of the manual to do with programming are available as a separate manual now:</td>
</tr>
<tr>
<td></td>
<td>• &quot;mvIMPACT_Acquire_API_CPP_manual.chm&quot;,</td>
</tr>
<tr>
<td></td>
<td>• &quot;mvIMPACT_Acquire_API_C_manual.chm&quot;, and</td>
</tr>
<tr>
<td></td>
<td>• &quot;mvIMPACT_Acquire_API_NET_manual.chm&quot;. These manuals can be download from <a href="http://www.matrix-vision.com">http://www.matrix-vision.com</a>.</td>
</tr>
<tr>
<td>30. September 2012</td>
<td>Moved Working with the Hardware Real-Time Controller (HRTC) (p. 181) to Use cases (p. 140).</td>
</tr>
<tr>
<td>20. September 2012</td>
<td>Added chapter &quot;Porting existing code written with versions earlier then 3.0.0&quot;.</td>
</tr>
<tr>
<td>17. August 2012</td>
<td>Added use case Adjusting sensor -x02d (-1012d) (p. 167).</td>
</tr>
<tr>
<td>21. June 2012</td>
<td>Added description, how to install the Linux driver using the installer script (Installing the mvIMPACT Acquire driver (p. 33)).</td>
</tr>
<tr>
<td>21. June 2012</td>
<td>Added information (electrical characteristic, pinning (p. 60)) about LVTTL version (mvBlueFOX-MLC2xxx-XLW).</td>
</tr>
<tr>
<td>02. April 2012</td>
<td>Enhanced chapter Output sequence of color sensors (RGB Bayer) (p. 74) and added chapter Bilinear interpolation of color sensors (RGB Bayer) (p. 75).</td>
</tr>
<tr>
<td>17. February 2012</td>
<td>Renewed chapter wxPropView (p. 78).</td>
</tr>
<tr>
<td>09. November 2011</td>
<td>Added Settings behaviour during startup (p. 43) in chapter Quickstart (p. 24).</td>
</tr>
<tr>
<td>26. July 2011</td>
<td>Removed chapter EventHandling. See &quot;Porting existing code written with versions earlier then 2.0.0&quot;.</td>
</tr>
<tr>
<td>11. July 2011</td>
<td>Added chapters &quot;Callback demo&quot;.</td>
</tr>
<tr>
<td>08. July 2011</td>
<td>Added chapter Using 2 mvBlueFOX-MLC cameras in Master-Slave mode (p. 175).</td>
</tr>
<tr>
<td>06. June 2011</td>
<td>Added chapters &quot;Porting existing code written with versions earlier then 2.0.0&quot;.</td>
</tr>
<tr>
<td>31. May 2011</td>
<td>Added chapter Creating double acquisitions (HRTC) (p. 185).</td>
</tr>
<tr>
<td>18. January 2011</td>
<td>Added chapter Setting up multiple display support and/or work with several capture settings in parallel (p. 95).</td>
</tr>
<tr>
<td>19. October 2010</td>
<td>Added chapters &quot;Chunk data format&quot;.</td>
</tr>
<tr>
<td>22. Sep. 2010</td>
<td>Added suitable for mvBlueFOX-MLC What’s inside and accessories (p. 22).</td>
</tr>
</tbody>
</table>
02. Aug. 2010 | mvBlueFOX-200W and mvBlueFOX-MLC100W support flash control output: CMOS sensors (p. 73)
Added chapter Import and Export images (p. 94).


19. Apr. 2010 | Added new example ContinuousCaptureDirectX.

01. Apr. 2010 | Added Use cases (p. 140) about high dynamic range (p. 163) of sensor mvBlueFOX-(Model)200w (0.4 Mpix [752 x 480]) (p. 232).


13. Jan. 2010 | Added chapters "Porting existing code written with versions earlier then 1.12.0".

11. Jan. 2010 | Due to a software update, documentation of CMOS sensor (-x00w) (p. 232) updated.

10. Nov. 2009 | Added Windows 7 as supported operating system.


22. Sep. 2009 | Added Wide-VGA sensor (p. 232) and removed sensor -x02.


05. May 2009 | Added figures which shows "how to connect flash to digital output".

05. May 2009 | Added book Use cases (p. 140), which offers solutions and explanations for standard use cases.

22. Jan. 2009 | Added information about how to test the general trigger functionality of the camera Setting up external trigger and flash control (p. 108).


09. Apr. 2008 | Corrected Figure 4: DIG OUT mvBlueFOX-1xx in Dimensions and connectors (p. 46).

25. Feb. 2008 | Added note about EEPROM of mvBlueFOX-M in Dimensions and connectors (p. 52).


01. Oct 2007 | Update sensor data in chapter Order code nomenclature (p. 19).

20. Aug. 2007 | Added part number of JST connectors used on the mvBlueFOX-M (see: Dimensions and connectors (p. 52).

31. Jul. 2007 | Rewritten "How to use this manual". This book now includes a getting started chapter (see: Composition of the manual (p. 2)).


29. May 2007 | Added an attention in chapter Quickstart (p. 24) section Installing the hardware (p. 27) (Windows) and Installing the hardware (p. 39) (Linux).

23. May 2007 | Added calculators to calculate the frame rate of the sensors (see specific sensor documentation: Sensor overview (p. 71)).

23. Apr. 2007 | Updated sensor description and added description of Micron’s CMOS 1280x1024 (-102a) (p. 235) sensor.

02. Apr. 2007 | Updated description of mvBlueFOX-M1xx digital I/O in chapter Dimensions and connectors (p. 52).

29. Jan. 2007 | Repainted Digi/O images (see: Dimensions and connectors (p. 46)).

24. Nov. 2006 | Added attention to the Digi/O description of the mvBlueFOX-M (see: Dimensions and connectors (p. 52).
<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>14. Nov. 2006</td>
<td>Updated Linux installation documentation (see: Quickstart (p.24)).</td>
</tr>
<tr>
<td>20. Oct 2006</td>
<td>Updated Linux installation documentation (see: Quickstart (p.24)).</td>
</tr>
<tr>
<td>11. Sep. 2006</td>
<td>Devided the Quickstart chapter into Linux® and Windows® (see: Quickstart (p.24)).</td>
</tr>
<tr>
<td>8. Sep. 2006</td>
<td>Updated CCD timing in CCD 640 x 480 (1/3&quot;) documentation (see: mvBlueFOX-[Model]220a (0.3 Mpix [640 x 480]) (p.200)).</td>
</tr>
<tr>
<td>5. Sep. 2006</td>
<td>Updated the sensor data (see: Sensor overview (p.71)).</td>
</tr>
<tr>
<td>23. Aug. 2006</td>
<td>Added general tolerance of the housing (see: Technical data (p.46)).</td>
</tr>
<tr>
<td></td>
<td>Added ambient temperature of the mvBlueFOX standard version (see: Components (p.51).</td>
</tr>
<tr>
<td>07. Jun. 2006</td>
<td>Extended the HRTC documentation (see: How to use the HRTC (p.125)).</td>
</tr>
<tr>
<td>02. Jun. 2006</td>
<td>Fixed image errors in CCD 640 x 480 (1/3&quot;) documentation (see: mvBlueFOX-[Model]220a (0.3 Mpix [640 x 480]) (p.200)).</td>
</tr>
<tr>
<td>01. Jun. 2006</td>
<td>Updated the chm index.</td>
</tr>
<tr>
<td>18. May 2006</td>
<td>Sensor description: Changed black/white to gray scale (see: Sensor overview (p.71)).</td>
</tr>
<tr>
<td>14. Feb. 2006</td>
<td>Added CCD 640 x 480 (1/3&quot;) (see: mvBlueFOX-[Model]220a (0.3 Mpix [640 x 480]) (p.200)).</td>
</tr>
<tr>
<td>13. Feb. 2006</td>
<td>Corrected the image of the &quot;4-pin circular plug-in connector&quot; (see: Dimensions and connectors (p.46)).</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

<table>
<thead>
<tr>
<th>![Warning]</th>
<th>We cannot and do not take any responsibility for the damage caused to you or to any other equipment connected to the mvBlueFOX. Similarly, warranty will be void, if a damage is caused by not following the manual.</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Warning]</td>
<td>Handle the mvBlueFOX with care. Do not misuse the mvBlueFOX. Avoid shaking, striking, etc. The mvBlueFOX could be damaged by faulty handling or shortage.</td>
</tr>
<tr>
<td>![Warning]</td>
<td>Use a soft cloth lightly moistened with a mild detergent solution when cleaning the camera.</td>
</tr>
<tr>
<td>![Warning]</td>
<td>Never face the camera towards the sun. Whether the camera is in use or not, never aim at the sun or other extremely bright objects. Otherwise, blooming or smear may be caused.</td>
</tr>
<tr>
<td>![Warning]</td>
<td>Please keep the camera closed or mount a lens on it to avoid the CCD or the CMOS from getting dusty.</td>
</tr>
<tr>
<td>![Warning]</td>
<td>Clean the CCD/CMOS faceplate with care. Do not clean the CCD or the CMOS with strong or abrasive detergents. Use lens tissue or a cotton tipped applicator and ethanol.</td>
</tr>
<tr>
<td>![Warning]</td>
<td>Never connect two USB cables to the mvBlueFOX even if one is only connected to a PC.</td>
</tr>
<tr>
<td>![Warning]</td>
<td>The mvBlueFOX is bus powered &lt; 2.5 W.</td>
</tr>
<tr>
<td>![Warning]</td>
<td>The mvBlueFOX meets IP40 standards.</td>
</tr>
<tr>
<td>![Warning]</td>
<td>Using the single-board or board-level versions:</td>
</tr>
<tr>
<td></td>
<td>• Handle with care and avoid damage of electrical components by electrostatic discharge (ESD):</td>
</tr>
<tr>
<td></td>
<td>– Discharge body static (contact a grounded surface and maintain contact).</td>
</tr>
<tr>
<td></td>
<td>– Avoid all plastic, vinyl, and styrofoam (except antistatic versions) around printed circuit boards.</td>
</tr>
<tr>
<td></td>
<td>– Do not touch components on the printed circuit board with your hands or with conductive devices.</td>
</tr>
</tbody>
</table>
5.1 High-Speed USB design guidelines

If you want to make own High-Speed (HS) USB cables, please pay attention to following design guidelines:

- Route High-Speed (HS) USB signals with a minimum number of vias and sharp edges!
- Avoid stubs!
- Do not cut off power planes VCC or GND under the signal line.
- Do not route signals no closer than 20 * h to the copper layer edge if possible (h means height over the copper layer).
- Route signal lines with 90 Ohm ± 15% differential impedance.
  - 7.5 mil printed circuit board track with 7.5 mil distance result in approx. 90 Ohm @ 110 um height over GND plane.
  - There are other rules when using double-ply printed circuit boards.
- Be sure that there is 20 mil minimum distance between High-Speed USB signal pair and other printed circuit board tracks (optimal signal quality).

5.2 European Union Declaration of Conformity statement

The mvBlueFOX complies with the provision of the following European Directives:

- 2014/30/EU (EMC directive)
- 2014/35/EU (LVD - low voltage directive)
- For EN 61000-6-3:2007, mvBlueFOX-IGC with digital I/O needs the Steward snap-on ferrite 28A0350-0B2 on I/O cable.
- For EN 61000-6-3:2007, mvBlueFOX-MLC with digital I/O needs the Würth Elektronik snap-on ferrite WE74271142 on I/O cable and copper foil on USB.

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.
Konformitätserklärung
Declaration of conformity

Der Hersteller
The Manufacturer

Matrix Vision GmbH
Talstraße 16
71570 Oppenweiler
Germany

erklärt hiermit, daß sein Produkt
hereby declares that her product

mvBlueFOX

den Bestimmungen der EU-EMV-Richtlinie 89/336 EWG
und des EMVG vom 9.11.1992 in folgenden Normen entsprechen:
correspond to the EU-EMC-Guideline 89/336/EEC
and the EMC law dated 9.11.1992 regarding following standards.

EN 55024 A : 1998
EN 61000-6-1

Bitte beachten:
Um die Einhaltung obiger Normen sicherzustellen, ist die Verwendung hochwertiger,
vollständig geschirrter Anschlusskabel unbedingt erforderlich.

Please note:
In order to fulfill the above standards, the use of high-quality shielded connecting cables is
required.

Oppenweiler, 18.05.2005

[Signature]
Dipl. Ing. (FH) Gerhard Thullner
Geschäftsführer / Manager
EG-Konformitätserklärung

Declaration of conformity

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

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

Typbezeichnung: mvBlueFOX-IGC

Type: mvBlueFOX-IGC

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

2014/30/EU (EMV-Richtlinie) / 2014/35/EU (EMC Directive)
auf Grundlage folgender harmonisierter Normen:
based on the following harmonized standards:

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

Störempfindlichkeit / Interference immunity
EN 61000-6-2 / 2005

Oppenweiler, 24.11.2015

i.V. Uwe Hagmaier
Leiter Entwicklung / Head of Development

Bitte beachten: Um die Einhaltung obiger Normen sicherzustellen, ist die Verwendung
hochwertiger, vollständig geschirmter Anschlusskabel unbedingt erforderlich.
Please note: in order to fulfill the above standards, the use of high-quality, shielded connecting
cables is required.
EG-Konformitätserklärung

Declaration of conformity

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

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

Typbezeichnung: mvBlueFOX-MLC
Type: mvBlueFOX-MLC

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

2014/30/EU (EMV-Richtlinie) / 2014/35/EU (EMC Directive)
auf Grundlage folgender harmonisierter Normen:
based on the following harmonized standards:

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

Störfestigkeit / Interference immunity
EN 61000-6-2 / 2005

Oppenweiler, 24.11.2015

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

Bitte beachten: Um die Einhaltung obiger Normen sicherzustellen, ist die Verwendung eines geeigneten EMV-Gehäuses sowie hochwertiger, vollständig geschirrter Anschlusskabel unbedingt erforderlich.
Please note: In order to fulfill the above standards, the use of a suitable EMC-case, as well as the use of high-quality shielded connecting cables is required.

5.3 Legal notice

5.3.1 For customers in the U.S.A.
This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a residential environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. However there is no guarantee that interferences will not occur in a particular installation. If the equipment does cause harmful interference to radio or television reception, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the distance between the equipment and the receiver.
- Use a different line outlet for the receiver.
- Consult a radio or TV technician for help.

You are cautioned that any changes or modifications not expressly approved in this manual could void your authority to operate this equipment. The shielded interface cable recommended in this manual must be used with this equipment in order to comply with the limits for a computing device pursuant to Subpart B of Part 15 of FCC Rules.

- To be compliant to FCC Class B, mvBlueFOX-IGC requires an I/O cable with an retrofittable ferrite to be used such as
  - Company: Steward Type: 28A0350-0B2

5.3.2 For customers in Canada

This apparatus complies with the Class B limits for radio noise emissions set out in the Radio Interference Regulations.

5.3.3 Pour utilisateurs au Canada

Cet appareil est conforme aux normes classe B pour bruits radioélectriques, spécifiées dans le Règlement sur le brouillage radioélectrique.
6 Introduction

The mvBlueFOX is a compact industrial CCD & CMOS camera solution for any PC with a Hi-Speed USB (USB 2.0) port. A superior image quality makes it suited for most applications. Integrated preprocessing like binning reduces the PC load to a minimum. The standard Hi-Speed USB interface guarantees an easy integration without any additional interface board. To make the cameras flexible to any industrial applications, the image processing tools mvIMPACT as well as different example solutions are available.

![mvBlueFOX Camera](image)

**Figure 1: mvBlueFOX**

The mvBlueFOX is suitable for following tasks:

- machine vision
- robotics
- surveillance
- microscopy
- medical imaging

With the name mvBlueFOX-M1xx, the industrial camera mvBlueFOX is also available as a single-board version.

6.1 Order code nomenclature

6.1.1 mvBlueFOX

The mvBlueFOX nomenclature scheme is as follows:

mvBlueFOX - A B - (1) (2) (3) (4)

- A: Sensor model
  - 220: 0.3 Mpix, 640 x 480, 1/4", CCD
  - 220a: 0.3 Mpix, 640 x 480, 1/3", CCD
  - 200w: 0.4 Mpix, 752 x 480, 1/3", CMOS
  - 221: 0.8 Mpix, 1024 x 768, 1/3", CCD
  - 202a: 1.3 Mpix, 1280 x 1024, 1/2", CMOS
  - 223: 1.4 Mpix, 1360 x 1024, 1/2", CCD
  - 224: 1.9 Mpix, 1600 x 1200, 1/1.8", CCD
  - 205: 5.0 Mpix, 2592 x 1944, 1/2.5", CMOS

- B: Sensor color
  - G: Gray scale version
  - C: Color version
6.1.2 mvBlueFOX-M

The mvBlueFOX-M nomenclature scheme is as follows:

mvBlueFOX-M A B - (1) (2) (3) (4)
- A: Sensor model
  220: 0.3 Mpix, 640 x 480, 1/4", CCD
  220a: 0.3 Mpix, 640 x 480, 1/3", CCD
  200w: 0.4 Mpix, 752 x 480, 1/3", CMOS
  221: 0.8 Mpix, 1024 x 768, 1/3", CCD
  202a: 1.3 Mpix, 1280 x 1024, 1/2", CMOS
  223: 1.4 Mpix, 1360 x 1024, 1/2", CCD
  224: 1.9 Mpix, 1600 x 1200, 1/1.8", CCD
  205: 5.0 Mpix, 2592 x 1944, 1/2.5", CMOS
- B: Sensor color
  G: Gray scale version
  C: Color version
- (1): Lensholder
  1: No holder (standard)
  2: C-mount with adjustable backfocus
  3: CS-mount with adjustable backfocus
  4: S-mount #9031
  5: S-mount #9033
- (2): Filter
  1: None (standard)
  2: IR-CUT
  3: Glass
  4: Daylight cut
- (3): Misc
  1: None (standard)
- (4): Misc
  1: None (standard)

6.1.3 mvBlueFOX-IGC

The mvBlueFOX-IGC nomenclature scheme is as follows:
6.1 Order code nomenclature

mvBlueFOX-IGC A B - (1) (2) (3) (4)

- A: Sensor model
  200w: 0.4 Mpix, 752 x 480, 1/3", CMOS
  202b: 1.2 Mpix, 1280 x 960, 1/3", CMOS
  202d: 1.2 Mpix, 1280 x 960, 1/3", CMOS
  202a: 1.3 Mpix, 1280 x 1024, 1/2", CMOS
  205: 5.0 Mpix, 2592 x 1944, 1/2.5", CMOS

- B: Sensor color
  G: Gray scale version
  C: Color version

- (1): Lensholder
  1: CS-mount without adjustable backfocus (standard)
  2: C-mount without adjustable backfocus (CS-mount with add. 5 mm extension ring)
  3: C-mount with adjustable backfocus

- (2): Filter
  1: IR-CUT (standard)
  2: Glass
  3: Daylight cut
  4: none

- (3): Case
  1: Color blue (standard)
  2: Color black, no logo, no label MATRIX VISION
  9: None

- (4): I/O
  1: None (standard)
  2: With I/O #08727

6.1.4 mvBlueFOX-MLC

The mvBlueFOX-MLC nomenclature scheme is as follows:

mvBlueFOX-MLC A B - C D E - (1) (2) (3) (4)

- A: Sensor model
  200w: 0.4 Mpix, 752 x 480, 1/3", CMOS
  202b: 1.2 Mpix, 1280 x 960, 1/3", CMOS
  202d: 1.2 Mpix, 1280 x 960, 1/3", CMOS
  202a: 1.3 Mpix, 1280 x 1024, 1/2", CMOS
  205: 5.0 Mpix, 2592 x 1944, 1/2.5", CMOS

- B: Sensor color
  G: Gray scale version
  C: Color version

- C: Mini USB
  U: with Mini USB (standard)
  X: without Mini USB

- D: Digital I/Os
  O: 1x IN + 1x OUT opto-isolated (standard)
  T: 2x TTL IN + 2x TTL OUT
  L: 3x LVTTL IN

- E: Connector
  W: board-to-wire (standard)
  B: board-to-board

- (1): Lensholder
  1: No holder (standard)
  2: C-mount with adjustable backfocus (CS-mount with add. 5 mm extension ring)
  3: C-mount with adjustable backfocus
  4: C-mount without adjustable backfocus
  5: CS-mount without adjustable backfocus
6: LENSHOLDER SH01F08V3 #09851  
7: LENSHOLDER SH02M13V2 #09951  
8: LENSHOLDER SH03H16V2 #09850

- (2): Filter  
  1: None (standard)  
  2: IR-CUT  
  3: Glass  
  4: Daylight cut

- (3): Misc  
  1: None (standard)

- (4): Misc  
  1: None (standard)

Examples:

<table>
<thead>
<tr>
<th>Model</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>mvBlueFOX-120G(^1)</td>
<td>640 x 480, CCD 1/4&quot;, gray</td>
</tr>
<tr>
<td>mvBlueFOX-102C(^1)</td>
<td>1280 x 1024, CMOS 1/2&quot;, color</td>
</tr>
<tr>
<td>mvBlueFOX-M121C(^1)</td>
<td>1024 x 768, CCD 1/3&quot;, color, module</td>
</tr>
<tr>
<td>mvBlueFOX-MLC200wC-XOW-5111</td>
<td>752 x 480, CMOS 1/3&quot;, color, single-board, without Mini-USB, 1x IN + 1x OUT opto-isolated, board-to-wire, CS-mount (w/o backfocus adjustment)</td>
</tr>
</tbody>
</table>

\(^1\): -1111 is the standard delivery variant and for this reason it is not mentioned.

6.2 What's inside and accessories

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

- mvBlueFOX
- instruction leaflet

For the first use of the mvBlueFOX we recommend the following accessories to get the camera up and running:

- A USB 2.0 cable

**Attention**

According to the customer and if the mvBlueFOX-MLC is shipped without lensholder, the mvBlueFOX-MLC will be shipped with a protective foil on the sensor. Before usage, please remove this foil!

Accessories for the mvBlueFOX:
### 6.2 What’s inside and accessories

<table>
<thead>
<tr>
<th>Part code</th>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ADAPTER CS-MOUNT</td>
<td>Lens fixing for mvBlueFOX to match with CS-mount lenses</td>
<td></td>
</tr>
<tr>
<td>KS-USB2-AA-EXT 05.0</td>
<td>USB 2.0 extension, active USB2 A plug to USB2 A jack, length 5m</td>
<td></td>
</tr>
<tr>
<td>KS-USB2-AB 03.0 TR</td>
<td>USB 2.0 cable A-B, transparent, Profi Line. Length 3m</td>
<td></td>
</tr>
<tr>
<td>KS-USB2-B4ST 02.0</td>
<td>USB 2.0 cable for mvBlueFOX, Binder 4pol to USB2-A. Length: 2m</td>
<td></td>
</tr>
<tr>
<td>KS-USB2-B4ST 03.0</td>
<td>USB 2.0 cable for mvBlueFOX, Binder 4pol to USB2-A. Length: 3m</td>
<td></td>
</tr>
<tr>
<td>KS-USB2-B4ST 05.0</td>
<td>USB 2.0 cable for mvBlueFOX, Binder 4pol to USB2-A. Length: 5m</td>
<td></td>
</tr>
<tr>
<td>KS-USB2-PHR4 01.5</td>
<td>USB connector cable for mvBlueFOX-M1xx. Length: 1.5m</td>
<td></td>
</tr>
<tr>
<td>KS-PHR12 500</td>
<td>Cable for mvBlueFOX-M1xx dig. I/O, 12-pin. Length: 500mm</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1..4</td>
<td>brown</td>
</tr>
<tr>
<td></td>
<td>5..8</td>
<td>gray</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>red</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>black</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>yellow</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>black</td>
</tr>
<tr>
<td>KS-MLC-IO-TTL 00.5</td>
<td>mvBlueFOX-MLC board-to-board TTL IO cable for master-slave synchronization (Molex plug to Molex plug). Length: 0.5m</td>
<td></td>
</tr>
<tr>
<td>KS-MLC-IO-W</td>
<td>mvBlueFOX-MLC board-to-wire I/O data cable, Molex 0510211200 with crimp terminal 50058. Length: up to 1m</td>
<td></td>
</tr>
<tr>
<td>KS-MLC-USB2-IO-W</td>
<td>mvBlueFOX-MLC board-to-wire I/O data and USB 2.0 cable, Molex 0510211200 with crimp terminal 50058 to USB2-A. Length: up to 1m</td>
<td></td>
</tr>
<tr>
<td>MV-Lensholder BFM-C</td>
<td>C-mount lensholder for mvBlueFOX-M, incl. IR-Cut filter</td>
<td></td>
</tr>
<tr>
<td>MV-Lensholder BFM-S 9031</td>
<td>S-mount lensholder M12 x 0.5 type MS-9031 for mvBlueFOX-M102</td>
<td></td>
</tr>
<tr>
<td>MV-Lensholder BFM-S 9033</td>
<td>S-mount lensholder M12 x 0.5 type: MS-9033 for mvBlueFOX-M102</td>
<td></td>
</tr>
<tr>
<td>MV-LENSHOLDER SH02M13</td>
<td>S-mount lensholder M12 x 0.5, height 13mm for mvBlueFOX-MLC</td>
<td></td>
</tr>
<tr>
<td>MV-LENSHOLDER SH01F08</td>
<td>S-mount lensholder M12 x 0.5, height 8mm for mvBlueFOX-MLC</td>
<td></td>
</tr>
<tr>
<td>ADAPTER S-C AD01S</td>
<td>Adapter for S-mount lens (M12x0,5) to C-mount, high penetration depth for mvBlueFOX-IGC</td>
<td></td>
</tr>
<tr>
<td>ADAPTER S-C AD02F</td>
<td>Adapter for S-mount lens (M12x0,5) to CS-mount, penetration depth: 5.5mm, outside diameters: 31mm for mvBlueFOX-IGC</td>
<td></td>
</tr>
<tr>
<td>MV-Tripod Adapter BF</td>
<td>Tripod adapter for mvBlueFOX</td>
<td></td>
</tr>
</tbody>
</table>
7 Quickstart

7.1 Windows

7.1.1 System Requirements

Currently supported Windows versions are:

- Microsoft Windows 7 (32-bit, 64-bit)
- Microsoft Windows 8.1 (32-bit, 64-bit)
- Microsoft Windows 10 (32-bit, 64-bit)

Other Windows version can be used at the user's own risk.

Note

Since mvIMPACT Acquire version 2.8.0 it could be possible that you have to update your Windows installer at least using Windows XP. The necessary packages are available from Microsoft's website: https://www.microsoft.com/en-US/download/details.aspx?id=8483

All necessary drivers are available from the MATRIX VISION website at www.matrix-vision.de, section "Products -> Cameras -> your interface -> your product -> Downloads".

Note

For Windows XP Embedded

As mvBlueFOX cameras will register as an 'imaging device' in the systems device manager please make sure that your Windows XP Embedded (XPe) distribution is shipped/build with support for the corresponding device class ("Class GUID {6bdd1fc6-810f-11d0-bec7-08002be2092f}") before installing the mvBlueFOX device driver. Otherwise, an installing of the driver when connecting a camera will fail with an error message like "a required section in the INF file could not be found". The camera will not be accessible then.

The mvBlueFOX is a USB 2.0 compliant camera device and needs therefore a functioning USB 2.0 port. If you are not sure about this, please follow these steps:

1. Press "Start" and click on "Run"
2. Enter msinfo32
3. Click on "Components" and look after "USB"
4. If there is a entry like "USB 2.0 Root Hub" or "ROOT_HUB20", your system has USB 2.0.

Please be sure that your system has at least one free USB port.
7.1 Windows

7.1.2 Installing the mvIMPACT Acquire driver

**Warning**

Before connecting the mvBlueFOX, please install the software and driver first!

All necessary drivers are available from the MATRIX VISION website: https://www.matrix-vision.com "Products -> Hardware -> mvBlueFOX -> Downloads Tab".

By double clicking on "mvBlueFOX-x86-n.n.n.msi" (for 32-bit systems) or "mvBlueFOX-x86_64-n.n.n.msi" (for 64-bit systems), the mvBlueFOX installer will start automatically.

![Figure 1: mvBlueFOX installer - Start window](image)

Select the folder, where you want to install the software.

![Figure 2: mvBlueFOX installer - Select folder](image)
Select the features 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.

- "mvBlueFOX driver"
  This is also not selectable.

- "Tools"
  This feature contains tools for the mvBlueFOX (e.g. to configure MATRIX VISION devices (mvDeviceConfigure) or to acquire images (wxPropView)).

- "Developer API"
  The Developer 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 makefiles 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 mvBlueFOX manual a single HTML help file (.chm).

![mvBlueFOX installer - Select features](image)

Figure 3: mvBlueFOX installer - Select features

Confirm the installation by clicking "Next".
7.1 Windows

Figure 4: mvBlueFOX installer - Confirm installation

The installation is finished now you can close the window.

Figure 5: mvBlueFOX installer - Finished installation

7.1.3 Installing the hardware
Warning

Before connecting the mvBlueFOX, please install the software and driver first!

It is not necessary to shutdown your system. On an USB port, it is possible to hot plug any USB device (hot plug lets you plug in new devices and use them immediately).

Warning

If using the Binder connector first connect the cable to the camera, then connect the camera to the PC.

Plug the mvBlueFOX to an USB 2.0 Port. After plugging the mvBlueFOX Windows® shows "Found New Hardware" and starts the Windows Hardware Wizard.

![Figure 6: Windows - Found new hardware](image)

The Wizard asks you for the driver. The installation doesn't need any Windows® automatic at this step and it is recommended to type the driver directory by hand. Choose "No, not this time" and press "Next".

![Figure 7: Windows Hardware Wizard - Driver Installation](image)

Choose "Install the software automatically" and press "Next".
The Hardware Wizard installs the driver.

The Hardware Wizard will search the registry for the device identification and after a while the Wizard prompts you to continue the installation or to abort it. Also Windows® will display the following message to inform the user that this driver digitally signed by Microsoft. You have to select 'Continue anyway' otherwise, the driver can't be installed. If you don't want to install a driver that is not signed you must stop the installation but can't work with the mvBlueFOX camera then.

Press "Continue Anyway" and finish the driver installation.
After the Windows® Logo testing, you have to click "Finish" to complete the installation.

Now, you can find the installed mvBlueFOX in the Windows® "Device Manager" under image devices.
After this installation, you can acquire images with the mvBlueFOX. Simply start the application **wxPropView** (p. 78) (wxPropView.exe) from mvBlueFOX/bin.

See also **wxPropView** (p. 78)

### 7.2 Linux

#### 7.2.1 System Requirements

**Kernel requirements**

Kernel 2.6.x .. Kernel 3.x.x.

- usbsfs support (CONFIG_USBDEVICEFS)

  **Note**
  
  This is different from devfs! support. The USB device file system should, of course, be mounted at /proc/bus/usb.

- SysV IPC support (CONFIG_SYSVIPC).

  **Note**
  
  Most distributions will have these kernel options turned on by default.
7.2.1.1 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 tool chain.

**Note**

Our own modified version of libusb has been statically linked to our library and is therefore included, so libusb is not a requirement.

7.2.1.2 Other requirements

- libexpat ([http://expat.sourceforge.net](http://expat.sourceforge.net))
- Optional: wxWidgets 2.6.x (non Unicode) for the wxWidget test programs.
- Optional: udev or hotplug subsystem (see also 4. 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" and "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.

7.2.1.3 Hardware requirements

**USB 2.0 Host controller (Hi-Speed) or USB 1.1 Host controller will also work (with a max. frame rate of 3 to 4 fps at 640x480 only).**

**Note**

We have noticed problems with some USB chip sets. At high data rates sometimes the image data appears to be corrupted. If you experience this you could try one or more of the following things.

- a different PC.
- a plug-in PCI/USB-2.0 card without any cables between the card and the USB connector.
- turning off the image footer property - this will ignore data errors.

**Note**

The driver contains libraries for Linux x86 (32 bit) or Linux 64-bit (x86_64). There are separate package files for systems with tool chains 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 tool chain is, the less likely it is that it will work. Tool chains based on GNU gcc 2.x.x are not supported at all.

GCC 4.x (4.1.0) has been tested on OpenSuSE 10.1 and should work on other platforms.

This version (32-bit only) will also run in a VMware ([http://www.vmware.com](http://www.vmware.com)) virtual machine!
7.2 Linux

7.2.2 Installing the mvIMPACT Acquire driver

To use the mvBlueFOX camera 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 mvBlueFOX camera, 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:

mvBlueFOX-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 (available as download from https://www.matrix-vision.com) 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_mvBlueFOX.sh /
   . && cp /media/cdrom/drv/Linux/mvBlueFOX-x86_ABI2-1.12.45.tgz -t ./

3. Run the install script:

   ~workspace$ ./install_mvBlueFOX.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_mvBlueFOX.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_mvBlueFOX.sh") and the archive ("mvBlueFOX-x86_ABI2-n.n.n.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

The **target directory name** specifies where to place the driver. If the directory does not yet exist, it will be created. The path can be either absolute or relative; i.e. the name may but need not start with "/.".

MATRIX VISION GmbH
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-1.12.45
```

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 mvBlueFOX-x86_AB←I2-n.n.n.tgz found in the current directory will be installed.

You can now start wxPropView (p.78), after installing the hardware (p.39) like

```
wxPropView
```

because the installer script added the needed symbolic links.

Note

If you want to install the mvBlueFOX Linux driver without installer script manually, please have a look at the following chapter:

7.2.2.1 Installing the mvIMPACT Acquire driver manually

Note

We recommend to use the installer script to install the mvBlueFOX driver (p.33).

The mvBlueFOX is controlled by a number of user-space libraries. It is not necessary to compile kernel modules for the mvBlueFOX.

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:

<table>
<thead>
<tr>
<th>Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.x.x (obsolete)</td>
<td>You cannot use the mvBlueFOX on your computer. Upgrade to a newer distribution.</td>
</tr>
<tr>
<td>3.2.x - 3.3.x (obsolete)</td>
<td>Use the C++ ABI 1. This package has ABI1 in its name.</td>
</tr>
<tr>
<td>greater or equal 3.4.x</td>
<td>Use the C++ ABI 2. This package has ABI2 in its name.</td>
</tr>
</tbody>
</table>
3. You can now install the mvBlueFOX libraries as follows:

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

The mvBlueFOX libraries are supplied as a "tgz" archive with the extension ".tgz". The older "autopackage" format is now deprecated since it cannot handle 64-bit libraries.

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

```bash
tar xvzf mvBlueFOX-x86_ABI2-1.12.45.tgz
```

**Note**

Current versions of the ABI1 libraries were compiled using a SuSE 8.1 system for maximum compatibility with older Linux distributions. These libraries should work with all SuSE 8.x and SuSE 9.x versions as well as with Debian Sarge and older Red Hat / Fedora variants.

Current versions of the ABI2 libraries were compiled using a SuSE 10.1 system for maximum compatibility with newer Linux distributions. These libraries should work with SuSE 10.x as well as with Ubuntu 6.06 or newer, with up-to-date Gentoo or Fedora FC5.

(b) After installing the mvBlueFOX 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 lib
-rw-r--r-- 1 root root 1079 Jan 5 15:08 makefile
```

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 other directories contain headers needed to write applications for the mvBlueFOX.

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":

**Note**

Please declare the device e.g. BF* or BF00001

```bash
cd my_mvbfo_directory
LD_LIBRARY_PATH='pwd'/lib/x86 apps/LiveSnap/x86/LiveSnap BF*
```

For 64-bit it will look like this...

```
LD_LIBRARY_PATH='pwd'/lib/x86_64 apps/LiveSnap/x86_64/LiveSnap BF*
```

For ARM it will look like this...

```
LD_LIBRARY_PATH='pwd'/lib/arm apps/LiveSnap/arm/LiveSnap BF*
```

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.
(c) 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.

(d) It may be possible to run applications as a non-root user on your system if you are using the udev system or a fairly recent version of hotplug.

7.2.2.2 For udev (e.g. Gentoo ∗, OpenSuSE 10.0 - 10.1)

Add the following 2 rules to one of the files in the directory "/etc/udev/rules.d" or make a new file in this directory (e.g. "/etc/udev/rules.d/20-mvbf.rules") containing the lines below:

```
ENV{UDEV_D_EVENT}="1", ACTION="add", BUS="usb", ENV{PRODUCT}="164c/101/0", /RUN+="/bin/sh -c 'chmod 0664 $env{DEVICE}; chgrp usb $env{DEVICE}'"
ENV{UDEV_D_EVENT}="1", ACTION="add", BUS="usb", ENV{PRODUCT}="164c/103/0", /RUN+="/bin/sh -c 'chmod 0664 $env{DEVICE}; chgrp usb $env{DEVICE}'"
```

You will find an example file "20-mvbf.rules" in the scripts directory after installation.

**Note**

Do not forget to add your user to the usb group! You may have to create the group first.

Current Gentoo systems support udev with a minimal, legacy hotplug system. It is usually sufficient to add any usernames that are to be used for the mvBlueFOX to the group "usb" because there is already a udev rule giving write permission to all USB devices to members of this group. If this does not work then try the other alternatives described here. The udev method is better because hotplug is likely to be removed eventually.

7.2.2.3 For udev (OpenSuSE 10.2 - 10.x)

In "/etc/fstab" in the line starting with usbfs change noauto to defaults

Afterwards either restart the system or execute "mount -a"

In "/etc/udev/rules.d/50-udev-default.rules" (or similar) change the line after a comment regarding libusb. Change 'MODE="0644"' in 'MODE="0664", GROUP="users"'

Connect the camera to the system now or re-connect it if it has been connected already

In case the environment variable USB_DEVFS_PATH is not set, it needs to be set to "/dev/bus/usb" ("export USB_DEVFS_PATH="/dev/bus/usb"."

---

MATRIX VISION GmbH
7.2.2.4 For udev on Ubuntu 06.10

Edit the file /etc/udev/rules.d/40-permissions.rules. Search for the entry for usbfs. It should look like this:

```
# USB devices (usbfs replacement)
SUBSYSTEM="usb_device" MODE="0664"
```

Now change it to read like this:

```
# USB devices (usbfs replacement)
SUBSYSTEM="usb_device" GROUP="plugdev", MOD="0664"
```

7.2.2.5 udev on some other systems

Some very up-to-date systems also set the environment variable `$USB_DEVFS_PATH` to point to `"/dev/bus/usb"` instead of the older (default) value of `"/proc/bus/usb"`. This may cause the mvBlueFOX libraries to attempt to access device nodes in `"dev/bus/usb"` but the rules described above will not change the permissions on these files. Normally you will find a rule in `"/etc/udev/rules.d/50-udev.rules"` which will already cure this problem. You might like to slightly modify this rule to give write permission to a specific group e.g. the group "usb". A patch is supplied in the scripts directory to do this.

7.2.2.6 Alternatively, for a system with full hotplugging (e.g. older SuSE systems)

Copy the files named below from the scripts directory into the directory `"/etc/hotplug/usb"`:

```
matrixvision.usermap
matrixvision_config
```

The file "matrixvision.usermap" contains the vendor and product IDs for the mvBlueFOX cameras and specifies that the script "matrixvision_config" should be run when a MATRIX VISION mvBlueFOX camera is plugged in or out. This script attempts to write some information to the system log and then changes the permissions for the newly-created device node so that non-root users can access the camera.

This feature has not yet been extensively tested. If you find that the applications start but appear to hang or to wait for a long time before continuing (and normally crashing) then changing the file permissions on your system does not appear to be sufficient. We have observed this on a 32 bit SuSE 9.1 system. In this case you may have more success if you change the owner of the application to "root" and set the suid bit to allow it to run with "root" permissions.

\[\text{Note}\]

This is considered a security risk by some experts.
If you have been using the mvBlueFOX with one user (e.g. root) and want to try it as another user you should remove the complete directory "/tmp/mv/", which may contain several files of zero length. These files are used to control mutexes within the software and will be owned by the first user. The new user will probably not be able to write to these files and the mvBlueFOX will not work.

MATRIX VISION GmbH
7.2.2.6.1 Using CMOS versions of the mvBlueFOX and mvBlueFOX-M especially with USB 1.1

Version 1.4.5 contains initial support for CMOS mvBlueFOX on USB 1.1. In order to conform to the rigid timing specifications of the CMOS sensor, onboard RAM is used. This RAM is available only on mvBlueFOX-M boards at the moment. Therefore you cannot use the mvBlueFOX-102x with USB 1.1. It will work with USB 2.0.

**Note**

If you want to capture continuous live images from mvBlueFOX-102 or mvBlueFOX-M102x you should switch the trigger mode from "Continuous" to "OnDemand" for best reliable results. For single snaps the default values should work correctly.

7.2.2.6.2 Power Down Mode

To try out the new power down mode you will need to update the firmware on the mvBlueFOX to at least version 39. There is a simple text-based utility to do this in "apps/FirmwareUpgrade". Just build the application and start on the platform of your choice giving the serial number of the connected mvBlueFOX as a parameter. E.g. on an Embedded ARM system:

```
$ ./arm/FirmwareUpgrade BF000188
++ Start FirmwareUpgrade: Sep 28 2006/11:48:19
Have found 1 devices on this platform!
  0 Serial: BF000188
Initialising the device: BF000188. This might take some time...
The firmware of device BF000188 is currently 37.
It will now be updated. During this time(approx. 30 sec.) the application will not react. Please be patient.
Successfully performed firmware update
Update successful.
Please disconnect and reconnect the device now to activate the new firmware.
Press any key to end the application
```

To test the power down mode use the test application in "apps/PowerDownTest". Take a look at the source code to see which function is used to switch on power down mode. Here is an example of the output for an Embedded ARM system:

```
$ ./arm/PowerDownTest BF000188
++ Start PowerDownTest sample: Sep 28 2006/11:48:43
Have found 1 devices on this platform!
  0 Serial: BF000188
Initialising the device: BF000188. This might take some time...
Ready to snap. Press ’p’<return> to power down, ’q’<return> to quit or /
<return> to snap an image..
Request Nr.: 0. Snap took 0.76s.,
Will save the file as: SingCapt.pgm
Ready to snap. Press ’p’<return> to power down, ’q’<return> to quit or /
<return> to snap an image..
Request Nr.: 1. Snap took 0.41s.,
Will save the file as: SingCapt.pgm
Ready to snap. Press ’p’<return> to power down, ’q’<return> to quit or /
<return> to snap an image..
Request Nr.: 2. Snap took 0.4s.,
Will save the file as: SingCapt.pgm
Ready to snap. Press ’p’<return> to power down, ’q’<return> to quit or /
<return> to snap an image..
p
Power off!
```
7.3 Relationship between driver, firmware and FPGA file

To operate a mvBlueFOX device apart from the physical hardware itself 3 pieces of software are needed:

- a firmware running on the device (provides low-level functionality like allowing the device to act as a USB device, support for multiple power states etc.)
- an FPGA file loaded into the FPGA inside the device (provides access features to control the behaviour of the image sensor, the digital I/Os etc.)
- a device driver (this is the mvBlueFOX.dll on Windows® and the libmvBlueFOX.so on Linux) running on the host system (provides control over the device from an application running on the host system)

The physical mvBlueFOX device has a firmware programmed into the device’s non-volatile memory, thus allowing the device to act as a USB device by just connecting the device to a free USB port. So the firmware version that will be used when operating the device does NOT depend on the driver version that is used to communicate with the device.

On the contrary the FPGA file version that will be used will be downloaded in volatile memory (RAM) when accessing the device through the device driver thus the API. One or more FPGA files are a binary part of the device driver. This shall be illustrated by the following figure:
As it can be seen in the image one or multiple firmware files are also a binary part of the device driver. However it is important to notice that this firmware file will **NOT** be used automatically but only when the user or an application explicitly updates the firmware on the device and will only become active after power-cycling the device. Since mvIMPACT Acquire version 2.28.0 every firmware starting from version 49 is available within a single driver library and can be selected for updating! mvDeviceConfigure however will always update the device firmware to the latest version. If you need to downgrade the firmware for any reason please get into contact with the MATRIX VISION support to get detailed instructions on how to do that.

7.3.1 FPGA

Until the device gets initialized using the API no FPGA file is loaded in the FPGA on the device. Only by opening the device through the API the FPGA file gets downloaded and only then the device will be fully operational:

As the FPGA file will be stored in RAM, disconnecting or closing the device will cause the FPGA file to be lost. The firmware however will remain:

In case multiple FPGA files are available for a certain device the FPGA file that shall be downloaded can be selected by an application by changing the value of the property `Device/CustomFPGAFileSelector`. However the value of this property is only evaluated when the device is either initialized using the corresponding API function **OR** if a device has been unplugged or power-cycled while the driver connection remains open and the device is then plugged back in.
7.3 Relationship between driver, firmware and FPGA file

Note

There is just a limited set of devices that offer more than one FPGA file and these additional FPGA files serve very special purposes so in almost every situation the default FPGA file will be the one used by an application. Before using custom FPGA files, please check with MATRIX VISION about why and if this makes sense for your application.

So assuming the value of the property Device/CustomFPGAFileSelector has been modified while the device has been unplugged, a different FPGA file will be downloaded once the device is plugged back into the host system:

![Figure 16: A different FPGA file can be downloaded](image)

7.3.2 Firmware

Only during a firmware update the firmware file that is a binary part of the device driver will be downloaded permanently into the device's non-volatile memory.

Warning

Until mvIMPACT Acquire 2.27.0 each device driver just contained one specific firmware version thus once a device's firmware has been updated using a specific device driver the only way to change the firmware version will be using another device driver version for upgrading/downgrading the firmware again. Since mvIMPACT Acquire version 2.28.0 every firmware starting from version 49 is available within a single driver library and can be selected for updating! mvDeviceConfigure however will always update the device firmware to the latest version. If you need to downgrade the firmware for any reason please get into contact with the MATRIX VISION support to get detailed instructions on how to do that.

So assume a device with a certain firmware version is connected to a host system:

![Figure 17: A certain firmware version is connected to a host system](image)

During an explicit firmware update, the firmware file from inside the driver will be downloaded onto the device. In order to become active the device must be power-cycled.
Figure 18: Firmware file will be downloaded during an firmware update...

When then re-attaching the device to the host system, the new firmware version will become active:

Figure 19: ... after repowering the device it will be active

- The current firmware version of the device can be obtained either by using one of the applications which are part of the SDK such as mvDeviceConfigure \[^{[117]}\] or by reading the value of the property Device/FirmwareVersion or Info/FirmwareVersion using the API.

- The current FPGA file version used by the device can be obtained by reading the value of the property Info/Camera/SensorFPGAVersion.

Using wxPropView the same information is available as indicated by the following figure:
7.4 Settings behaviour during startup

A setting contains 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 analogue to digital conversion process for analogue 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:
Figure 21: wxPropView - Device setting start 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). E.g. you have a device 'VD000001' which belongs to the product group 'VirtualDevice' with a setting exclusively for 'VD000001'. As soon as you store a product specific setting, the (device specific) setting for 'VD000001' will be deleted. 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. 78) 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. 78) 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. 78) 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. wxPropView (p. 78) 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 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 Power supply

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Comment</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$U_{USBPOWER_{IN}}$</td>
<td>mvBlueFOX power supply via USB</td>
<td>4.75</td>
<td>5</td>
<td>5.25</td>
<td>V</td>
</tr>
<tr>
<td>$I_{USBPOWER_{IN}}@ 5V / 40MHz$</td>
<td></td>
<td>280</td>
<td>500</td>
<td>mA</td>
<td></td>
</tr>
<tr>
<td>$I_{USBPOWER_{IN}}$ (Power Off Mode - only with mvBlueFOX-IGC / mvBlueFOX-MLC)</td>
<td></td>
<td>66</td>
<td>mA</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8.2 Standard version (mvBlueFOX-xxx)

8.2.1 Dimensions and connectors

Figure 1: Connectors mvBlueFOX

<table>
<thead>
<tr>
<th>front view</th>
<th>side view</th>
<th>back view</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Front View" /></td>
<td><img src="image2.png" alt="Side View" /></td>
<td><img src="image3.png" alt="Back View" /></td>
</tr>
</tbody>
</table>

Figure 2: Dimensional drawing of tripod adapter

8.2.1.1 D-Sub 9-pin (male)
Figure 3: D-Sub 9-pin (male), digital I/O
Pin | Signal | Description
--- | --- | ---
1 | IN0- | Negative terminal of opto-isolated input 

2 | OUT0- | Negative terminal of opto-isolated output (emitter of npn-phototransistor)

3 | OUT1- | Negative terminal of opto-isolated output (emitter of npn-phototransistor)

4 | IN1- | Negative terminal of opto-isolated input *

5 | N/C | 

6 | IN0+ | Positive terminal of opto-isolated input *

7 | OUT0+ | Positive terminal of opto-isolated output (collector of npn-phototransistor)

8 | OUT1+ | Positive terminal of opto-isolated output (collector of npn-phototransistor)

9 | IN1+ | Positive terminal of opto-isolated input *

1 Voltage between + and - may be up to 26V, input current is 17mA.

8.2.1.1 Characteristics of the digital inputs

Open inputs will be read as a logic zero.

When the input voltage rises above the trigger level, the input will deliver a logic one.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Comment</th>
<th>Min.</th>
<th>Std.</th>
<th>Max.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>( U_{IN_TTL} )</td>
<td>High level input voltage TTL logic</td>
<td>3</td>
<td>5</td>
<td>6.5</td>
<td>V</td>
</tr>
<tr>
<td>( U_{IN_TTL} )</td>
<td>Low level input voltage TTL logic</td>
<td>-0.7</td>
<td>1</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>( I_{IN_TTL} )</td>
<td>Current TTL logic</td>
<td>8.5</td>
<td>12</td>
<td>mA</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Comment</th>
<th>Min.</th>
<th>Std.</th>
<th>Max.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>( U_{IN_PLC} )</td>
<td>High level input voltage PLC logic</td>
<td>12</td>
<td>24</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>( U_{IN_PLC} )</td>
<td>Low level input voltage PLC logic</td>
<td>-0.7</td>
<td>8</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>( I_{IN_PLC} )</td>
<td>Current PLC logic</td>
<td>17</td>
<td>25</td>
<td>mA</td>
<td></td>
</tr>
</tbody>
</table>

Figure 4: DigIn mvBlueFOX-xxx

In wxPropView (p. 78) you can change between

- **TTL** ("DigitalInputThreshold = 2V") and
- **PLC** ("DigitalInputThreshold = 10V")
input behavior of the digital inputs using the DigitalInputThreshold property in "Digital I/O -> DigitalInputThreshold".

![Image](wxPropView-DigitalInputThreshold.png)

**Figure 5: wxPropView - DigitalInputThreshold**

### 8.2.1.1.2 Characteristics of the digital outputs

<table>
<thead>
<tr>
<th></th>
<th>( U_{\text{min}} ) [V]</th>
<th>( U_{\text{max}} ) [V]</th>
<th>( I_{\text{min}} ) [mA]</th>
<th>( I_{\text{max}} ) [mA]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>30</td>
<td>100</td>
<td>100 (on state current)</td>
<td></td>
</tr>
</tbody>
</table>

![Image](DigOut-mvBlueFOX-xxx.png)

**Figure 6: DigOut mvBlueFOX-xxx**

### 8.2.1.3 Connecting flash to digital output

You can connect a flash in series to the digital outputs as shown in the following figure, however, you should only use LEDs together with a current limiter:
Figure 7: Connecting flash (LEDs) to DIG OUT

8.2.1.2 USB connector, type B (USB 2.0)

Figure 8: USB B connector (female)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>USBPOWER_IN</td>
</tr>
<tr>
<td>2</td>
<td>D-</td>
</tr>
<tr>
<td>3</td>
<td>D+</td>
</tr>
<tr>
<td>4</td>
<td>GND</td>
</tr>
<tr>
<td>Shell</td>
<td>shield</td>
</tr>
</tbody>
</table>

**Note**
The mvBlueFOX is an USB device!

**Attention**
Do not connect both USB ports at the same time.

8.2.1.3 4-pin circular plug-in connector with lock (USB 2.0)

Figure 9: 4-pin circular plug-in connector (female)
8.2 Standard version (mvBlueFOX-xxx)

### Pin Configuration

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal 'R' version</th>
<th>Signal 'U' version</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>USBPOWER_IN</td>
<td>Power out from USB</td>
</tr>
<tr>
<td>2</td>
<td>D+</td>
<td>not connected</td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
<td>GND</td>
</tr>
<tr>
<td>4</td>
<td>D-</td>
<td>not connected</td>
</tr>
</tbody>
</table>

Manufacturer: Binder  
Part number: 99-3390-282-04

**Note**

Differentiation between 'R' and 'U' version is obsolete. New mvBlueFOX versions have both connectors (circular connector and standard USB). The pin assignment corresponds to the description of 'R' version. While mvBlueFOX is connected and powered via standard USB, it is possible to connect additional power via circular connector *(only power; the data lines must be disconnected!)*. Only in this case, the power switch will change the power supply, if the current entry via standard USB is equal to or under the power supply of "circular connector".

**Attention**

Do not connect both USB ports at the same time.

8.2.2 LED states

<table>
<thead>
<tr>
<th>State</th>
<th>LED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera is not connected or defect</td>
<td>LED off</td>
</tr>
<tr>
<td>Camera is connected and active</td>
<td>Green light on</td>
</tr>
</tbody>
</table>

8.2.3 Components

- FPGA for image processing
- pixel clock up to 40 MHz
- reliable image transfer
  - using bulk-mode
  - image data surrounded by headers
- several trigger modes
  - auto, SW, external
- flash control output
  - using opto-isolated outputs
- opto-isolated I/O
  - 2 inputs, 2 outputs on D-Sub 9 connector
- bus powered
  - no external power supply needed
• two USB connectors
  – standard USB or circular plug-in connector 4 pin locked
• ambient temperature operation: 0..45 deg C / 30..80 RH
• ambient temperature storage: -20..60 deg C / 20..90 RH

Additional features of mvBlueFOX-2xx:

• 8 Mega-pixel image memory (FiFo)
• new ADC
• 10 Bit mode

8.3 Board-level version (mvBlueFOX-Mxxx)

8.3.1 Dimensions and connectors

Figure 10: mvBlueFOX-M12x (CCD) with C-mount
The **mvBlueFOX-M** has a serial I²C bus EEPROM with 64 KBit of which 512 Bytes can be used to store custom arbitrary data.
See also

UserDataEntry class description

8.3.1.1 4-pin Wire-to-Board header (USB 2.0)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Comment</th>
<th>Cable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>USBPOWER_IN</td>
<td>Supply voltage</td>
<td>red</td>
</tr>
<tr>
<td>2</td>
<td>USB_DATA-</td>
<td>Data</td>
<td>white</td>
</tr>
<tr>
<td>3</td>
<td>USB_DATA+</td>
<td>Data</td>
<td>green</td>
</tr>
<tr>
<td>4</td>
<td>GND</td>
<td>Ground</td>
<td>black</td>
</tr>
</tbody>
</table>

Manufacturer: JST
Part number: B4B-PH-K

8.3.1.2 12-pin Wire-to-Board header (Dig I/O)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FPGA_IO0</td>
<td>Digital In 0</td>
</tr>
<tr>
<td>2</td>
<td>FPGA_IO1</td>
<td>Digital In 1</td>
</tr>
<tr>
<td>3</td>
<td>FPGA_IO2</td>
<td>Digital In 2</td>
</tr>
<tr>
<td>4</td>
<td>FPGA_IO3</td>
<td>Digital In 3</td>
</tr>
<tr>
<td>5</td>
<td>FPGA_IO4</td>
<td>Digital Out 0</td>
</tr>
<tr>
<td>6</td>
<td>FPGA_IO5</td>
<td>Digital Out 1</td>
</tr>
<tr>
<td>7</td>
<td>FPGA_IO6</td>
<td>Digital Out 2</td>
</tr>
<tr>
<td>8</td>
<td>FPGA_IO7</td>
<td>Digital Out 3</td>
</tr>
<tr>
<td>9</td>
<td>MAINPOWER</td>
<td>Current from the USB cable</td>
</tr>
<tr>
<td>10</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>11</td>
<td>VCC24V</td>
<td>24 V output (10mA)</td>
</tr>
<tr>
<td>12</td>
<td>GND</td>
<td>Ground</td>
</tr>
</tbody>
</table>

Manufacturer: JST
Part number: B12B-PH-K

Attention

Do not connect Dig I/O signals to the FPGA pins until the mvBlueFOX-M has been started and configured. Otherwise, you will risk damaging the mvBlueFOX-M hardware!
8.3 Board-level version (mvBlueFOX-Mxxx) 55

See also

High-Speed USB design guidelines (p. 14)

8.3.1.3 Contact

Figure 13: Contact, dimensions in mm (in.)

<table>
<thead>
<tr>
<th>Application wire</th>
<th>Q'ty / reel</th>
</tr>
</thead>
<tbody>
<tr>
<td>mm²</td>
<td>AWG #</td>
</tr>
<tr>
<td>0.05 to 0.22</td>
<td>30 to 24</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Material and finish: phosphor bronze, tin-plated
Manufacturer: JST
Part number: SPH-002T-P0.5S

8.3.1.4 Housing

Figure 14: Housing, dimensions in mm (in.)

<table>
<thead>
<tr>
<th>Circuits</th>
<th>Dimensions in mm (in.)</th>
<th>Q'ty / box</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>4</td>
<td>6.0 (.236)</td>
<td>9.8 (.386)</td>
</tr>
<tr>
<td>12</td>
<td>22.0 (.866)</td>
<td>25.8 (1.016)</td>
</tr>
</tbody>
</table>

Material and finish: nylon 66, UL94V-0, natural (white)
Manufacturer: JST
Part number: PHR-4 / PHR-12

See also

Suitable assembled cable accessories for mvBlueFOX-M: What’s inside and accessories (p. 22)
8.3.1.5 Characteristics of the mvBlueFOX-Mxxx digital I/Os

8.3.1.5.1 Dig I/O max. values

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Comment</th>
<th>Min</th>
<th>Nom</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>U\textsubscript{DIG IN}</td>
<td>Input voltage</td>
<td>-3</td>
<td>0</td>
<td>3.6</td>
<td>V</td>
</tr>
</tbody>
</table>

8.3.1.5.2 Characteristics of the digital inputs

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Comment</th>
<th>Min</th>
<th>Nom</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>U\textsubscript{DIG IN LOW}</td>
<td>low level input voltage (I\textsubscript{IN} = 1.67mA)</td>
<td>-0.9</td>
<td>0</td>
<td>0.9</td>
<td>V</td>
</tr>
<tr>
<td>U\textsubscript{DIG IN HIGH}</td>
<td>high level input voltage (I\textsubscript{IN} = 1.67mA)</td>
<td>2.6</td>
<td>3.3</td>
<td>3.6</td>
<td>V</td>
</tr>
<tr>
<td>I\textsubscript{IN}</td>
<td>input current (@ 3.3V)</td>
<td>0.1</td>
<td>4</td>
<td>1.7</td>
<td>mA</td>
</tr>
</tbody>
</table>

8.3.1.5.3 Characteristics of the digital outputs

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Comment</th>
<th>Min</th>
<th>Nom</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>I\textsubscript{DIG OUT}</td>
<td>current at digital output</td>
<td>-12</td>
<td>+24</td>
<td>mA</td>
<td></td>
</tr>
<tr>
<td>U\textsubscript{DIG OUT HIGH}</td>
<td>digital output (I\textsubscript{OUT}=12mA)</td>
<td>1.6</td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Digital output (I\textsubscript{OUT}&lt;2mA)</td>
<td>2.6</td>
<td>3.4</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>U\textsubscript{DIG OUT LOW}</td>
<td>digital output (I\textsubscript{OUT}=2mA)</td>
<td>0.2</td>
<td>V</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[U_{\text{DIG OUT HIGH}} \text{ min} = 2.8 \cdot I_{\text{OUT}} \times 100\]
The Dig I/O are connected directly via a resistor to the FPGA pins and therefore they are not protected. For this reason, an application has to provide a protection circuit to the digital I/O of mvBlueFOX-M.

Note

The Dig I/O characteristics of the mvBlueFOX-M are not compatible to the Dig I/O of the mvBlueFOX standard version.

8.3.2 LED states

<table>
<thead>
<tr>
<th>State</th>
<th>LED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera is not connected or defect</td>
<td>LED off</td>
</tr>
<tr>
<td>Camera is connected and active</td>
<td>Green light on</td>
</tr>
</tbody>
</table>

8.3.3 Components

- 8 Mpixels image memory

8.3.4 Accessories mvBlueFOX-Mxxx

8.3.4.1 mvBlueFOX-M-FC-S

The mvBF-M-FC-S contains high capacity condensers with switching electronics for transferring stored energy of the condensers to external flash LEDs. It is possible to connect 2 pushbuttons/switches to the 8-pin header (CON3 - Control connector). Additionally, 2 LED interfaces are available. There are two version of mvBF-M-FC-S:

- Model 1 can be connected to mvBlueFOX-M with a cable via CON5.
- Model 2 can be mounted on the mvBlueFOX-M via CON1 directly.
8.3.4.1.1 CON2 - Flash connector

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flash +</td>
<td>Flash power</td>
</tr>
<tr>
<td>2</td>
<td>Flash -</td>
<td>Switched to ground (low side switch)</td>
</tr>
</tbody>
</table>
8.3 Board-level version (mvBlueFOX-Mxxx)

Manufacturer: JST
Part number: B-2B-PH

8.3.4.1.2 CON3 - Control connector

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>GND</td>
<td>LED2 cathode connector / board ground</td>
</tr>
<tr>
<td>2</td>
<td>LED2 output</td>
<td>LED2 anode connector</td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
<td>LED1 cathode connector / board ground</td>
</tr>
<tr>
<td>4</td>
<td>LED1 output</td>
<td>LED1 anode connector</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
<td>Board ground</td>
</tr>
<tr>
<td>6</td>
<td>Input2</td>
<td>Switch to ground for setting Input2</td>
</tr>
<tr>
<td>7</td>
<td>GND</td>
<td>Board ground</td>
</tr>
<tr>
<td>8</td>
<td>Input1</td>
<td>Switch to ground for setting Input1</td>
</tr>
</tbody>
</table>

Manufacturer: JST
Part number: B-8B-PH-SM4 TB

8.3.4.1.3 Electrical characteristic

<table>
<thead>
<tr>
<th>Signal</th>
<th>Parameter</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>GND</td>
<td>Board ground</td>
<td>0</td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LED 1/2 output (anode)</td>
<td>Output voltage</td>
<td>5</td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Internal series resistance</td>
<td>470</td>
<td>Ohm</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Forward current $I_F$ at $U_{LED} = 2V$</td>
<td>6</td>
<td>mA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input 1/2 (internal 10k pull up to 3.3V)</td>
<td>$V_{IL}$ (low level input voltage)</td>
<td>0.9</td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$V_{IH}$ (high level input voltage)</td>
<td>2.5</td>
<td>5.5</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Flash +</td>
<td>Voltage (open contact)</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>Flash output capacitance</td>
<td>528</td>
<td>uF</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Internal capacitance storage energy</td>
<td>0.190</td>
<td>Ws</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Flash capacitance charge current / output DC current</td>
<td>20</td>
<td>mA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flash ²</td>
<td>$I_{OUT}$</td>
<td>-2</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>On voltage at $I_{OUT MAX}$</td>
<td>0.15</td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Off voltage</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>V</td>
</tr>
</tbody>
</table>

1 Depends on mvBlueFOX-M power supply
2 Attention: No over-current protection!
8.4 Single-board version (mvBlueFOX-MLC2xx)

8.4.1 Typical Power consumption @ 5V

<table>
<thead>
<tr>
<th>Model</th>
<th>Power consumption (+/- 10%)</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>-200w</td>
<td>1.09</td>
<td>W</td>
</tr>
<tr>
<td>-202a</td>
<td>1.39</td>
<td>W</td>
</tr>
<tr>
<td>-202b</td>
<td>1.58</td>
<td>W</td>
</tr>
<tr>
<td>-202d</td>
<td>1.28</td>
<td>W</td>
</tr>
<tr>
<td>-205</td>
<td>1.37</td>
<td>W</td>
</tr>
</tbody>
</table>

8.4.2 Dimensions and connectors

<table>
<thead>
<tr>
<th>front view</th>
<th>side view</th>
<th>back view</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>1.6</td>
</tr>
<tr>
<td>without S-mount</td>
<td>35</td>
<td>30.4</td>
</tr>
<tr>
<td></td>
<td>32.4</td>
<td>24</td>
</tr>
</tbody>
</table>

Figure 20: mvBlueFOX-MLC (without S-mount)

**Note**

The mvBlueFOX-MLC has a serial I²C bus EEPROM with 16 KByte of which 8 KByte are reserved for the firmware and 8 KByte can be used to store custom arbitrary data.

**See also**

UserDataEntry class description
8.4.2.1 Sensor’s optical midpoint and orientation

The sensor’s optical midpoint is in the center of the board (Figure 21: intersection point of the holes diagonals). The (0,0) coordinate of the sensor is located at the one bottom left corner of the sensor (please notice that Mini-B USB connector is located at the bottom at the back).

**Note**

Using a lens, the (0,0) coordinate will be mirrored and will be shown at the top left corner of the screen as usual!

![Figure 21: Sensor's optical midpoint and orientation](image)

8.4.2.2 Mini-B USB (USB 2.0)

![Figure 22: Mini-B USB](image)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>USBPOWER_IN</td>
<td>Supply voltage</td>
</tr>
<tr>
<td>2</td>
<td>USB_DATA-</td>
<td>Data</td>
</tr>
<tr>
<td>3</td>
<td>USB_DATA+</td>
<td>Data</td>
</tr>
<tr>
<td>4</td>
<td>ID</td>
<td>Not connected</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
<td>Ground</td>
</tr>
</tbody>
</table>

8.4.2.3 12-pin Wire-to-Board header (USB 2.0 / Dig I/O)
If you have the mvBlueFOX-MLC variant which uses the standard Mini-B USB connector, pin 2 and 3 (USB_DATA+_ / USB_DATA-) of the header won't be connected!
<table>
<thead>
<tr>
<th>pin</th>
<th>Opto-isolated variant</th>
<th>TTL compliant variant</th>
<th>LVTTL compliant variant (only available for mvBlueFOX-MLC202aG)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>GND</td>
<td>GND</td>
<td>GND</td>
</tr>
<tr>
<td>2</td>
<td>USB_D←ATA+</td>
<td>USB_D←ATA+</td>
<td>USB_D←ATA+</td>
</tr>
<tr>
<td>3</td>
<td>USB_D←ATA-</td>
<td>USB_D←ATA-</td>
<td>USB_D←ATA-</td>
</tr>
<tr>
<td>4</td>
<td>USBPO←WER_IN</td>
<td>USBPO←WER_IN</td>
<td>USBPO←WER_IN</td>
</tr>
<tr>
<td>5</td>
<td>I2C SDA</td>
<td>I2C SDA</td>
<td>I2C SDA</td>
</tr>
<tr>
<td>6</td>
<td>I2C SCL</td>
<td>I2C SCL</td>
<td>I2C SCL</td>
</tr>
<tr>
<td>7</td>
<td>USBPO←WER_IN</td>
<td>USBPO←WER_IN</td>
<td>USBPO←WER_IN</td>
</tr>
<tr>
<td>8</td>
<td>GND</td>
<td>GND</td>
<td>GND</td>
</tr>
<tr>
<td>9</td>
<td>OUT0-</td>
<td>OUT</td>
<td>N.C.</td>
</tr>
<tr>
<td>10</td>
<td>OUT0+</td>
<td>OUT0</td>
<td>IN2</td>
</tr>
</tbody>
</table>

**Signal**

- **1** GND Ground
- **2** USB_D←ATA+ Data
- **3** USB_D←ATA- Data
- **4** USBPO←WER_IN Supply voltage
- **5** I2C SDA Serial data line (the I2C interface is master-only, which means that I2C slaves can only be connected externally)
- **6** I2C SCL Serial clock line (the I2C interface is master-only, which means that I2C slaves can only be connected externally)
- **7** USBPO←WER_IN Supply voltage
- **8** GND Ground
- **9** OUT0- Opto-isolated digital output 0 (Negative voltage)
- **10** OUT0+ Opto-isolated digital output 0 (Positive voltage)

**Comment**

- **1** Ground
- **2** Data
- **3** Data
- **4** Supply voltage
- **5** Serial data line
- **6** Serial clock line
- **7** Supply voltage
- **8** Ground
- **9** Opto-isolated digital output 0
- **10** Opto-isolated digital output 0
### 8.4.2.3.1 Electrical characteristic

#### Digital inputs TTL

**Note**

If the digital input is not connected, the state of the input will be “1” (as you can see in \textit{wxPropView} (p. 78)).
TTL compliant variant

<table>
<thead>
<tr>
<th>Comment</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_{IN}$</td>
<td>- 0.5 mA</td>
<td>- 0.5 mA</td>
<td>- 0.5 mA</td>
<td>mA</td>
</tr>
<tr>
<td>$U_{IN}$</td>
<td>$V_{IH}$: 3.8 V</td>
<td>5.5 V</td>
<td>6 V</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>$V_{IL}$: - 0.8 V</td>
<td>1.3 V</td>
<td>- 0.8 V</td>
<td>V</td>
</tr>
</tbody>
</table>

LVTTTL compliant variant

<table>
<thead>
<tr>
<th>Comment</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_{IN}$</td>
<td>- 0.5 mA</td>
<td>- 0.5 mA</td>
<td>- 0.5 mA</td>
<td>mA</td>
</tr>
<tr>
<td>$U_{IN}$</td>
<td>$V_{IH}$: 2 V</td>
<td>3.8 V</td>
<td>3 V</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>$V_{IL}$: - 0.8 V</td>
<td>0.8 V</td>
<td>- 0.8 V</td>
<td>V</td>
</tr>
</tbody>
</table>

TTL input low level / high level time: Typ. < 210ns

*Digital outputs TTL*

![TTL digital outputs block diagram](image)

**Figure 24: TTL digital outputs block diagram**

<table>
<thead>
<tr>
<th>Comment</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_{OUT}$</td>
<td>Dig_out power</td>
<td>+32 mA</td>
<td>+32 mA</td>
<td>mA</td>
</tr>
<tr>
<td>$U_{OUT}$</td>
<td>$V_{OH}$ (I_{OUT}=32mA): 3.8 V</td>
<td>8 V</td>
<td>8 V</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>$V_{OH}$: 5.25 V</td>
<td>5.25 V</td>
<td>5.25 V</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>$V_{OL}$ (I_{OUT}=32mA): 0.55 V</td>
<td>1 V</td>
<td>1 V</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>$V_{OL}$: 0.55 V</td>
<td>1 V</td>
<td>1 V</td>
<td>V</td>
</tr>
</tbody>
</table>

TTL output low level / high level time: Typ. < 40ns

*Opto-isolated digital inputs*
Delay

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Symbol</th>
<th>Typ.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turn-On time</td>
<td>$t_{\text{ON}}$</td>
<td>3</td>
<td>us</td>
</tr>
</tbody>
</table>

The inputs can be connected directly to +3.3 V and 5 V systems. If a higher voltage is used, an external resistor must be placed in series (Figure 25).

<table>
<thead>
<tr>
<th>Used input voltage</th>
<th>External series resistor</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3 V .. 5 V</td>
<td>none</td>
</tr>
<tr>
<td>12 V</td>
<td>680 Ohm</td>
</tr>
<tr>
<td>24 V</td>
<td>2 KOhm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$U_{\text{IN}}$</th>
<th>Comment</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{\text{IH}}$</td>
<td>3</td>
<td></td>
<td>5.5</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>$V_{\text{IL}}$</td>
<td>0.8</td>
<td>5</td>
<td>5</td>
<td>V</td>
<td></td>
</tr>
</tbody>
</table>

**Opto-isolated digital outputs**

Figure 25: Opto-isolated digital inputs block diagram with example circuit

Delay

Figure 26: Opto-isolated digital outputs block diagram with example circuit
8.4 Single-board version (mvBlueFOX-MLC2xx)

8.4.1 Characteristics

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Symbol</th>
<th>Test conditions</th>
<th>Typ.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turn-On time</td>
<td>$t_{ON}$</td>
<td>$R_L = 100$ Ohm, $V_{CC} = 10$V, $I_C = 2$mA</td>
<td>3</td>
<td>us</td>
</tr>
<tr>
<td>Storage time</td>
<td>$t_s$</td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Turn-Off time</td>
<td>$t_{OFF}$</td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Turn-On time</td>
<td>$t_{ON}$</td>
<td>$R_L = 1.9$ kOhm, $V_{CC} = 5$V, $I_C = 16$mA</td>
<td>2</td>
<td>us</td>
</tr>
<tr>
<td>Storage time</td>
<td>$t_s$</td>
<td></td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>Turn-Off time</td>
<td>$t_{OFF}$</td>
<td></td>
<td>40</td>
<td></td>
</tr>
</tbody>
</table>

8.4.3 LED states

- Camera is not connected or defect: LED off
- Camera is connected but not initialized or in "Power off" mode: Orange light on
- Camera is connected and active: Green light on

8.4.4 Assembly variants

The mvBlueFOX-MLC is available with following differences:

- Mini-B USB connector and digital I/O pin header
  - 1/1 opto-isolated or 2/2 TTL compliant digital I/O
- USB via header without Mini-B USB connector
- Female board connector instead of pin header (board-to-board connection)
- 3 different S-mount depths
• C(S)-mount compatibility using mvBlueCOUGAR-X flange
• ambient temperature operation: 5..55 deg C / 30..80 RH
• ambient temperature storage: -25..60 deg C / 20..90 RH

8.5 Single-board version with housing (mvBlueFOX-IGC2xx)

8.5.1 Dimensions and connectors

Figure 28: mvBlueFOX-IGC

<table>
<thead>
<tr>
<th>Lens protrusion</th>
<th>C-Mount</th>
<th>CS-Mount</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>10 mm</td>
<td>5 mm</td>
</tr>
</tbody>
</table>

Figure 29: mvBlueFOX-IGC-3xxx with adjustable backfocus

<table>
<thead>
<tr>
<th>Lens protrusion</th>
<th>C-Mount</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>8 mm (9.5 mm with max. Ø 20 mm)</td>
</tr>
</tbody>
</table>

Note
The mvBlueFOX-IGC has a serial I²C bus EEPROM with 16 KByte of which 8 KByte are reserved for the firmware and 8 KByte can be used to store custom arbitrary data.

See also
UserDataEntry class description
8.5 Single-board version with housing (mvBlueFOX-IGC2xx)

8.5.1 Mini-B USB (USB 2.0)

![Mini-B USB connector diagram]

**Figure 30: Mini-B USB**

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>USBPOWER_IN</td>
<td>Supply voltage</td>
</tr>
<tr>
<td>2</td>
<td>USB_DATA-</td>
<td>Data</td>
</tr>
<tr>
<td>3</td>
<td>USB_DATA+</td>
<td>Data</td>
</tr>
<tr>
<td>4</td>
<td>ID</td>
<td>Not connected</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
<td>Ground</td>
</tr>
</tbody>
</table>

8.5.1.2 4-pin circular plug-in connector with lock (I/O)

![4-pin circular connector diagram]

**Figure 31: 4-pin circular plug-in connector (female)**

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Comment</th>
<th>Color (of cable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IN0 +</td>
<td>Opto-isolated digital input 0 (Positive voltage)</td>
<td>brown</td>
</tr>
<tr>
<td>2</td>
<td>IN0 -</td>
<td>Opto-isolated digital input 0 (Negative voltage)</td>
<td>white</td>
</tr>
<tr>
<td>3</td>
<td>OUT0 +</td>
<td>Opto-isolated digital output 0 (Positive voltage)</td>
<td>blue</td>
</tr>
<tr>
<td>4</td>
<td>OUT0 -</td>
<td>Opto-isolated digital output 0 (Negative voltage)</td>
<td>black</td>
</tr>
</tbody>
</table>

Manufacturer: Binder
Part number: 79-3107-52-04

8.5.1.2.1 Electrical characteristic

Please have a look at the mvBlueFOX-MLC digital I/O characteristics (opto-isolated model) of the **12-pin Wire-to-Board Header (USB / Dig I/O)** (p. 60).

8.5.2 LED states

<table>
<thead>
<tr>
<th>State</th>
<th>LED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera is not connected or defect</td>
<td>LED off</td>
</tr>
<tr>
<td>Camera is connected but not initialized or in “Power off” mode.</td>
<td>Orange light on</td>
</tr>
<tr>
<td>Camera is connected and active</td>
<td>Green light on</td>
</tr>
</tbody>
</table>
8.5.3 Positioning tolerances of sensor chip

The sensor's optical midpoint is in the center of the housing. However, several positioning tolerances in relation to the housing are possible because of:

- Tolerance of mounting holes of the printed circuit board in relation to the edge of the lens holder housing is not specified but produced according to general tolerance DIN ISO 2768 T1 fine.
- Tolerance of mounting holes on the printed circuit board because of the excess of the holes ± 0.1 mm (Figure 32; 2).
- Tolerance between conductive pattern and mounting holes on the printed circuit board. Because there is no defined tolerance between conductive pattern and mounting holes, the general defined tolerance of ± 0.1 mm is valid (Figure 32; 1 in the Y-direction ± 0.1 mm; 3 in the Z-direction ± 0.1 mm).

There are further sensor specific tolerances, e.g. for model mvBlueFOX-IGC200wG:

- Tolerance between sensor chip MT9V034 (die) and its package (connection pad)
  - Chip position in relation to the mechanical center of the package: 0.2 mm (± 0.1mm) in the X- and Y-direction (dimensions in the sensor data sheet according to ISO 1101)
- Tolerance between copper width of the sensor package and the pad width of the printed circuit board
  During the soldering the sensor can swim to the edge of the pad: width of the pad 0.4 mm (possible tolerance is not considered), width of pin at least 0.35 mm, max. offset: ± 0.025mm

Further specific tolerances of other models on request.

![Figure 32: Positioning tolerances of sensor chip](image)

**Note**

There are also tolerances in lens which could lead to optical offsets.
9 Sensor overview

By default, the steps exposure and readout out of an image sensor are done one after the other. By design, CCD sensors support overlap capabilities also combined with trigger (see figure). In contrast, so-called pipelined CMOS sensors only support the overlapped mode. Even less CMOS sensors support the overlapped mode combined with trigger. Please check the sensor summary (p. 71). In overlapping mode, the exposure starts the exposure time earlier during readout.

9.1 CCD sensors

The CCD sensors are highly programmable imager modules which incorporate the following features:

<table>
<thead>
<tr>
<th>Sensors</th>
<th>0.3 Mpixels resolution CCD sensor (-220)</th>
<th>0.3 Mpixels resolution CCD sensor (-220a)</th>
<th>0.8 Mpixels resolution CCD sensor (-221)</th>
<th>1.4 Mpixels resolution CCD sensor (-223)</th>
<th>1.9 Mpixels resolution CCD sensor (-224)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor supplier</td>
<td>Sony</td>
<td>Sony</td>
<td>Sony</td>
<td>Sony</td>
<td>Sony</td>
</tr>
<tr>
<td>Sensor name</td>
<td>ICX098 AL/BL</td>
<td>ICX424 AL/AQ</td>
<td>ICX204 AL/AQ</td>
<td>ICX267 AL/AQ</td>
<td>ICX274 AL/AQ</td>
</tr>
<tr>
<td>Resolution</td>
<td>640 x 480 gray scale or RGB Bayer mosaic</td>
<td>640 x 480 gray scale or RGB Bayer mosaic</td>
<td>1024 x 768 gray scale or RGB Bayer mosaic</td>
<td>1360 x 1024 gray scale or RGB Bayer mosaic</td>
<td>1600 x 1200 gray scale or RGB Bayer mosaic</td>
</tr>
<tr>
<td>Sensor format</td>
<td>1/4&quot;</td>
<td>1/3&quot;</td>
<td>1/3&quot;</td>
<td>1/2&quot;</td>
<td>1/1.8&quot;</td>
</tr>
<tr>
<td>Pixel clock</td>
<td>12 MHz / 24 MHz</td>
<td>20 MHz / 40 MHz</td>
<td>20 MHz / 40 MHz</td>
<td>tbd / 40 MHz</td>
<td>20 MHz / 40 MHz</td>
</tr>
<tr>
<td>------------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------------</td>
<td>-------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Max. frames per second</td>
<td>60</td>
<td>100</td>
<td>39</td>
<td>20</td>
<td>16</td>
</tr>
<tr>
<td>Binning</td>
<td>H+V</td>
<td>H+V</td>
<td>H+V</td>
<td>H+V</td>
<td>H+V</td>
</tr>
<tr>
<td>Exposure time</td>
<td>44 us - 10 s</td>
<td>26 us - 10 s</td>
<td>44 us - 10 s</td>
<td>33 us - 10 s</td>
<td>30 us - 10 s</td>
</tr>
<tr>
<td>ADC (on sensor board) resolution</td>
<td>12 bit (up to 10 bit transmission)</td>
<td>12 bit (up to 10 bit transmission)</td>
<td>12 bit (up to 10 bit transmission)</td>
<td>12 bit (up to 10 bit transmission)</td>
<td>12 bit (up to 10 bit transmission)</td>
</tr>
<tr>
<td>Programmable analog gain and offset</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Frame integrating progressive scan sensor (no interlaced problems!)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High resolution</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High color reproductivity (for color version)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High sensitivity, low dark current</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Continuous variable-speed shutter</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pipelined in continuous / triggered mode</td>
<td>X / -</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low smear</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Excellent antiblooming characteristics</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programmable exposure time from usec to sec.</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programmable readout timing with free capture windows and partial scan</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger (Hardware / Software)</td>
<td>X / X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pipelined in continuous / triggered mode</td>
<td>X / -</td>
<td>X / -</td>
<td>X / -</td>
<td>X / -</td>
<td>X / -</td>
</tr>
<tr>
<td>Flash control output, synchronous to exposure period</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
With max. frame rate, image quality losings might be occur.

9.2 CMOS sensors

The CMOS sensor modules incorporate the following features:

<table>
<thead>
<tr>
<th>Sensors:</th>
<th>0.4 Mpix-resolution CMOS sensor (-200w)</th>
<th>1.3 Mpix-resolution CMOS sensor (-202a)</th>
<th>1.2 Mpix-resolution CMOS sensor (-202b)(^1)</th>
<th>1.2 Mpix-resolution CMOS sensor (-202d)(^1) only -MLC/-IGC</th>
<th>5.0 Mpix-resolution CMOS sensor (-205)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor supplier</td>
<td>Aptina</td>
<td>Aptina</td>
<td>Aptina</td>
<td>Aptina</td>
<td>Aptina</td>
</tr>
<tr>
<td>Sensor name</td>
<td>MT9V034</td>
<td>MT9M001</td>
<td>MT9M021</td>
<td>MT9M034</td>
<td>MT9P031</td>
</tr>
<tr>
<td>Resolution</td>
<td>752 x 480 gray scale or RGB Bayer mosaic</td>
<td>1280 x 1024 gray scale</td>
<td>1280 x 960 gray scale or RGB Bayer mosaic</td>
<td>1280 x 960 gray scale or RGB Bayer mosaic</td>
<td>2592 x 1944 gray scale or RGB Bayer mosaic</td>
</tr>
<tr>
<td>Indication of sensor category to be used</td>
<td>1/3&quot;</td>
<td>1/2&quot;</td>
<td>1/3&quot;</td>
<td>1/3&quot;</td>
<td>1/2.5&quot;</td>
</tr>
<tr>
<td>Pixel clock</td>
<td>40 MHz</td>
<td>40 MHz</td>
<td>40 MHz</td>
<td>40 MHz</td>
<td>40 MHz</td>
</tr>
<tr>
<td>Max. frames per second (in free-running full frame mode)</td>
<td>93(^2)</td>
<td>25</td>
<td>25</td>
<td>25</td>
<td>5.8</td>
</tr>
<tr>
<td>Binning</td>
<td>H+V (frame rate 170 Hz)</td>
<td>H+V, Average-()H+V (frame rate unchanged)</td>
<td>H+V, Average-()H+V (frame rate unchanged)</td>
<td>H+V, Average-()H+V (frame rate unchanged)</td>
<td>H+V, 3H+3V, AverageH+V, Average3H+3V, DroppingH+V, Dropping3-()H+3V (frame rate 22.7 Hz)</td>
</tr>
<tr>
<td>Exposure time</td>
<td>6 us - 4 s</td>
<td>100 us - 10 s</td>
<td>10 us - 4 s</td>
<td>10 us - 4 s</td>
<td>10 us - 10 s</td>
</tr>
<tr>
<td>ADC resolution</td>
<td>10 bit (10 / 8 bit transmission)</td>
<td>10 bit (10 / 8 bit transmission)</td>
<td>10 bit (10 / 8 bit transmission)</td>
<td>10 bit (10 / 8 bit transmission)</td>
<td>10 bit (10 / 8 bit transmission)</td>
</tr>
<tr>
<td>SNR</td>
<td>42 dB</td>
<td>40 dB</td>
<td>&lt; 43 dB</td>
<td>37.4 dB</td>
<td>37.4 dB</td>
</tr>
<tr>
<td>DR (normal / HDR)(^{p.163})</td>
<td>55 dB / &gt; 110 dB</td>
<td>61 dB / -</td>
<td>&gt; 61 dB /</td>
<td>&gt; 61 dB / with gray scale version</td>
<td>65 dB /</td>
</tr>
<tr>
<td>Progressive scan sensor (no interlaced problems!)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Rolling shutter</td>
<td>-</td>
<td>X</td>
<td>-</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Global shutter</td>
<td>X</td>
<td>-</td>
<td>X</td>
<td>-</td>
<td>X</td>
</tr>
</tbody>
</table>

\(^1\) With max. frame rate, image quality losings might be occur.
<table>
<thead>
<tr>
<th>Trigger (Hardware / Software)</th>
<th>X / X</th>
<th>X / -</th>
<th>X / -</th>
<th>X / -</th>
<th>X / X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pipelined in continuous / triggered mode</td>
<td>X / -</td>
<td>X / -</td>
<td>X / -</td>
<td>X / -</td>
<td>X / - (reset only)</td>
</tr>
<tr>
<td>High color reproductivity (for color version)</td>
<td>X</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Programmable readout timing with free capture windows and partial scan</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Flash control output, synchronous to exposure period</td>
<td>X</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>

**More specific data**

| mvBlueFO → X-[Model]200w (0.4 Mpix [752 x 480]) (p.232) | mvBlueFO → X-[Model]202a (1.3 Mpix [1280 x 1024]) (p.235) | mvBlueFO → X-[Model]202b (1.2 Mpix [1280 x 960]) (p.238) | mvBlueFO → X-[Model]202d (1.2 Mpix [1280 x 960]) (p.242) | mvBlueFO → X-[Model]205 (5.0 Mpix [2592 x 1944]) (p.245) |

---

1. The operation in device specific AEC/AGC mode is limited in (non continuous) triggered modes. AEC/AGC only works while trigger signal is active. When the trigger signal is removed AEC/AGC stops and gain and exposure will be set to a static value. This is due to a limitation of the sensor chip.

2. Frame rate increase with reduced AOI width, but only when width ≥ 560 pixels, below frame rate remains unchanged.

**Note**

For further information about rolling shutter, please have a look at the practical report about rolling shutter on our website:  [https://www.matrix-vision.com/tl_files/mv11/Glossary/art_rolling_shutter_en.pdf](https://www.matrix-vision.com/tl_files/mv11/Glossary/art_rolling_shutter_en.pdf)

For further information about image errors of image sensors, please have a look at Correcting image errors of a sensor (p.144).

### 9.3 Output sequence of color sensors (RGB Bayer)

![Readout Sequence Diagram](chart.png)
9.4 Bilinear interpolation of color sensors (RGB Bayer)

For Bayer demosaicing in the camera, we use bilinear interpolation:

![Bilinear Interpolation Diagram](image)

1. Interpolation of green pixels: the average of the upper, lower, left and right pixel values is assigned as the G value of the interpolated pixel.
   
   For example:
   
   \[
   G_8 = \frac{G_3 + G_7 + G_9 + G_{13}}{4}
   \]

   For G7:
   
   \[
   G_{7\_new} = 0.5 \times G_7 + 0.5 \times \frac{G_1 + G_3 + G_{11} + G_{13}}{4}
   \]

2. Interpolation of red/blue pixels:
   - Interpolation of a red/blue pixel at a green position: the average of two adjacent pixel values in corresponding color is assigned to the interpolated pixel.
     
     For example:
     
     \[
     B_7 = \frac{B_6 + B_8}{2} ; R_7 = \frac{R_2 + R_{12}}{2}
     \]
   - Interpolation of a red/blue pixel at a blue/red position: the average of four adjacent diagonal pixel values is assigned to the interpolated pixel.
     
     For example:
     
     \[
     R_8 = \frac{R_2 + R_4 + R_{12} + R_{14}}{4} ; B_{12} = \frac{B_6 + B_8 + B_{16} + B_{18}}{4}
     \]

   Any colored edge which might appear is due to Bayer false color artifacts.

**Note**

There are more advanced and adaptive methods (like edge sensitive ones) available if the host is doing this debayering.
10 Filters

MATRIX VISION offers two filters for the mvBlueFOX camera. The IR filter (p. 76) is part of the standard delivery condition.

10.1 Hot mirror filter

The hot mirror filter FILTER IR-CUT 15,5X1,75 FE has great transmission in the visible spectrum and blocks out a significant portion of the IR energy.

<table>
<thead>
<tr>
<th>Technical data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diameter</td>
</tr>
<tr>
<td>Thickness</td>
</tr>
<tr>
<td>Material</td>
</tr>
<tr>
<td>Characteristics</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>AOI</td>
</tr>
<tr>
<td>Surface quality</td>
</tr>
<tr>
<td>Surface irregularity</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Figure 1: FILTER IR-CUT 15,5X1,75 FE wavelengths and transmission diagram

10.2 Cold mirror filter

The FILTER DL-CUT 15,5X1,5 is a high-quality day light cut filter and has optically polished surfaces. The polished surface allows the use of the filter directly in the path of rays in image processing applications. The filter is protected against scratches during the transport by a protection film that has to be removed before installing the filter.

<table>
<thead>
<tr>
<th>Technical data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diameter</td>
</tr>
<tr>
<td>Thickness</td>
</tr>
<tr>
<td>Material</td>
</tr>
<tr>
<td>Characteristics</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
10.3 Glass filter

It is also possible to choose the glass filter "FILTER GLASS 15,5X1,75" with following characteristics:

<table>
<thead>
<tr>
<th>Technical data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Glass thickness</td>
</tr>
<tr>
<td>Material</td>
</tr>
<tr>
<td>Surface quality</td>
</tr>
<tr>
<td>Surface irregularity</td>
</tr>
</tbody>
</table>

Figure 2: FILTER DL-CUT 15,5X1,5 wavelengths and transmission diagram

Note

For further information how to change the filter, please have a look at our website:
11 Application Usage

11.1 wxPropView

wxPropView (p. 78) 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. 78)

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

wxPropView - Introduction:

11.1.1 How to work with wxPropView

wxPropView - Working with wxPropView:

Depending on the driver version, wxPropView starts with the Quick Setup Wizard (p. 78) (as soon as a camera with the right firmware version was selected used or a single camera with the right firmware was found) or without it (p. 81).

11.1.1.1 Quick Setup Wizard
Since
mvIMPACT Acquire 2.11.3

The Quick Setup Wizard is a tiny and powerful single window configuration tool to optimize the image quality automatically and to set the most important parameters, which affect the image quality, in an easy way manually and to get a preview of this changes. Settings will be accepted by clicking ok, otherwise the changes are cancelled.

![Quick Setup Wizard started](image)

Figure 1: Quick Setup Wizard started

Depending on the camera spectrum (gray or color sensor), it will automatically pre-set the camera so that image quality is usually as best as possible.

"For all cameras:"
Image format is chosen as 10 bit (if possible) as a good compromise on image quality and speed. It will further set

- "Exposure" to Auto,
- "Gain" to Auto,
- "Frame rate" to Auto based on current settings of the camera, and
- switches camera into continuous mode

"In case of gray:"
The above settings will be also applied whenever the "Gray Preset" button is pressed. For gray cameras it is herewith assumed that image processing prefers a linear camera response.

"In case of color:"
It will additionally set

- "White balance" in the camera to Auto, and will apply
- a host based moderate "Gamma correction" (1.8), and lastly it will apply
- a host (PC) based sensor specific "Color Correction Matrix" and use the respective "sRGB display matrix".

These settings will also be applied whenever the "Color Preset" button is pressed. It is herewith assumed that color camera image is optimized for best human visual feedback.
11.1.1.1.1 Changing the Presets

There are 3 presets:

- Gray
- Color
- Factory

Factory can be used as a fall back to quickly skip or remove all presets and load the factory default settings.

11.1.1.1.2 Modifying Settings

All auto modes can be switched off and all settings, such as Gain, Exposure etc. can be subsequently modified by using:

- the sliders,
- the arrow keys, or
- entering real values with your keyboard.

Toggling Gamma button loads or unloads a host based 10 bit Gamma correction with a moderate value of 1.8 into the signal processing path. Switch Gamma on if you require a gray level camera image to appear natural for the human eye.

Toggling Color+ button switches both CCM and sRGB display matrix on and off. This optimizes the sensor color response for the human eye and goes in conjunction with a display color response. Because sRGB displays are mostly used and this is the default color space in Windows OS, these are preselected. If you require other display matrices (e.g. Adobe or WideGamut) feel free to use the tree mode of wxPropView and select ColorTwistOutput Correction accordingly.

Setting Black Level
Black level can be used if you require dark portions in the image to appear even darker or brighter. Please note that this slider combines analog and digital settings meaningfully.

Setting Gain
Gain settings also combine analog and digital registers into one slider setting.

Setting Saturation
Saturation setting increases the color saturation to make the image appear more colored. It does not change uncolored parts in the image nor changes the color tone or hue.

11.1.1.1.3 How to disable Quick Setup Wizard

Uncheck the checkbox “Show This Display When A Device Is Opened” to disable the Quick Setup Wizard to be called automatically. Use the “Wizards” menu and select “Quick Setup” to open the Quick Setup Wizard once again.

11.1.1.1.4 How to Return to the Tree Mode

Use OK to use the values and settings of the Quick Setup Wizard and go back to the tree mode of wxPropView.

Use Cancel to discard the Quick Setup Wizard values and settings and go back to wxPropView and use the former (or default) settings.
11.1.1.5 Image Display Functions

Quick Setup Wizard allows zooming into the image by right clicking in the image area and unchecking "Fit To Screen" mode. Use the mouse wheel to zoom in or out. Check "Fit To Screen" mode, if you want the complete camera image to be sized in the window screen size.

11.1.1.6 Known Restrictions

In cases of Tungsten (artificial) light, camera brightness may tend to oscillations if Auto functions are used. This can be minimized or avoided by setting the frame frequency to an integer divisor of the mains frequency.

- Example:
  - Europe: 50 Hz; Set frame rate to 100, 50, 25 12.5 fps or appropriate.
  - In countries with 60 Hz use 120, 60, 30 or 15... accordingly.

11.1.2 First View of wxPropView

wxPropView (p.78) consists of several areas:

- "Menu Bar" (to work with wxPropView (p.78) 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.
11.1 wxPropView

11.1.1.3 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 mvBlueFOX camera as the capturing device.

For color sensors, it is recommended to perform a white balance (p. 104) calibration before acquiring images. This will improve the quality of the resulting images significantly.

---

Figure 3: wxPropView - First start
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

### 11.1.1.3.1 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.

![Figure 4: wxPropView - Using the record mode.](image-url)
11.1.1.3.2 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.

Figure 5: wxPropView - Hard Disk Recording.

11.1.1.4 Using the analysis plots

With the analysis plots you have the possibility to get image details and to export them (p. 93).
11.1.1.4.1 Spatial noise histogram

The spatial noise histogram calculates and evaluates statistically the difference between two neighboring 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.

![Figure 6: wxPropView - Spatial noise histogram](image)

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.

11.1.1.4.2 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 eliminated, 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.
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!

11.1.1.5 Storing and restoring settings

When wxPropView (p. 78) 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.
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 behaviour during startup (p. 43)

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 behaviour during startup (p. 43)

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.
11.1.1.6 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".

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:
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:
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:

![Figure 15: wxPropView - After applying the value range to a property](image)

**11.1.1.7 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 \texttt{TDMR\_ERROR} or \texttt{TPROPHANDLING\_ERROR}.

11.1.1.8 Copy grid data to the clipboard

Since \texttt{wxPropView} (p.\pageref{wxPropView}) 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.

![Figure 17: wxPropView - Copying grid data to the clipboard](image)
11.1.1.9 Import and Export images

**wxPropView** (p. 78) 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. 78) 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. 78) 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. 78)
- by starting **wxPropView** (p. 78) from the command line passing the file to open as a command line parameter (p. 116) (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. 78), 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.

![Figure 18: wxPropView - Raw image file import](image-url)
11.1.10 Setting up multiple display support and/or work with several capture settings in parallel

**wxPropView** (p. 78) 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.

The amount of parallel image displays can be configured via the **command line parameter** (p. 116) "dcx" and "dcy". In this step by step setup wxPropView (p. 78) 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".

![Image Display Configuration](image.png)

Figure 19: wxPropView - Create capture setting

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. 78) 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".

Figure 20: wxPropView - Create capture setting

2. Then, the user is asked for the name of the new setting.

Figure 21: wxPropView - Create capture setting - Choosing name

3. And finally for the base this new setting shall be derived from.
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. 78) 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)".

Figure 23: wxPropView - two settings

Figure 24: wxPropView - Assigning displays

Figure 25: wxPropView - Assigning displays
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:

![Figure 26: wxPropView - Setting up request count](image)

Finally, **wxPropView** 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** (p.78) 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.

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.

11.1.1.11 Bit-shifting an image

*wxPropView* 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.

---

Figure 30: Mid-grey 12 bit pixel image and 8 bit display with 2 example shifts
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} (p. 78) 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.

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

With \textit{wxPropView} (p. 78) 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.
11.1.13 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.

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.

See also

**Accessing log files using Linux** (p. 139)

11.1.2 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:
The next chapter will show how to set the interface layout and which interface you should use according to your needs.

11.1.2.1 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.

11.1.2.2 White balance of a camera device (color version)

Start the **wxPropView** (p. 78) and initialize the device by clicking "Use" and start a "Continuous" acquisition.

![Figure 33: wxPropView - Starting window](image)

While using a color version of the camera, the PC will calculate a color image from the original gray Bayer mosaic data. For getting correct colors when working with a Bayer mosaic filter you have to calibrate the white balance (this must be performed every time the lighting conditions change).

The **wxPropView** (p. 78) offers predefined settings for e.g.

- "Daylight",
- "TungstenLamp",
- "HalogenLamp",
- "FluorescentLamp" and many more.

Simply select the necessary item in the menu "**Image Settings** -> Base -> **ImageProcessing** -> **WhiteBalance**" ("DeviceSpecific interface layout") or "**Setting** -> Base -> **ImageProcessing** -> **WhiteBalance**" ("GenICam interface layout").

If you need a user defined setting, you can also define own ones. For this, select a profile (e.g. User1) for this setting:
Figure 34: wxPropView - Selecting WhiteBalance profile

**Note**

You can use up to 4 profiles.

Point the camera on a white or light gray area (the pixels do not have to be saturated, so use gray values between 150 and 230).

Go to the menu item "WhiteBalanceCalibration" and select the parameter "Calibrate Next Frame":

By committing the selected value, the application accepts the change. The next acquired image will be the reference for the white balance.

All further acquired images will be balanced with this setting:

For easier handling and easier working, all settings can be saved by clicking the menu Action -> Capture Settings -> Save ... (p. 87).
11.1.2.3 Configuring different trigger modes

To configure a device for a triggered acquisition, in wxPropView (p. 78) the property “Image Setting -> Camera -> TriggerMode” (“DeviceSpecific interface layout”) or “Setting -> Base -> Camera -> GenICam -> Acquisition Control -> Trigger Selector” (“GenICam interface layout”) is available.

Note

The supported trigger modes of each sensor are described in the More specific data (p. 71) of each sensor.

11.1.2.4 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.

Figure 37: wxPropView - Call refresh
11.1.2.5 Setting up external trigger and flash control

To set up external trigger and flash control, following things are required:

- mvBlueFOX with CCD sensor
- Host PC with USB 2.0 interface
- USB 2.0 cable with max. 5m
- Cable for supplying external trigger signal to camera
- Flash with needed cable and power supply
- Current mvBlueFOX driver

The camera is only connected by USB 2.0 cable to PC. All other signals are connected directly to the camera. Trigger and flash signals are directly controlled by the FPGA, which does the timing in the camera. This makes the trigger and flash control independent from CPU load of host PC or temporary USB 2.0 interrupts.

![Figure 38: mvBlueFOX with trigger and flash](image)

**Trigger control**
External trigger signal resets the image acquisition asynchronously to any other timing so that reaction delay is such short that it can be ignored. If a delay between trigger signal and starting integration is needed, it can be defined. By default it is set to 0 us.

**Flash control**
Signal for flash control is immediately set as soon as image integration begins. If a delay is needed it can be defined. By default this delay is set to 0.
11.1.2.5.1 Connection

External trigger signal
Signal for triggering image acquisition must be connected to digital input on backside of mvBlueFOX on D-Sub 9-pin connector (p. 46). You can choose either input IN0 (pin 6 and 1) or IN1 (pin 9 and 4) for triggering.

![Diagram of external trigger signal](image)

Figure 39: External trigger signal

Schematic shows how to fit application's switch to camera's digital input. External trigger signal must be in following conditions:

- **TTL (5 V):** High min. 3 V, Low max. 1 V
- **PLC (24 V):** High min. 12 V, Low max. 10 V

Application's switch can be a mechanical one any light barrier or some kind of encounter.

**Note**

Depending on used switch it might be necessary to use a pull-up or pull-down resistor so that camera input can recognize signal correctly.
To test the general trigger functionality, please follow these steps:

1. Select the "Acquisition Mode" "Continuous".
2. Click on "Acquire".
3. Set the "TriggerMode" to "OnHighLevel".
4. Set the "TriggerMode" to "DigitalInputThreshold" to e.g. 2V.
5. Connect a standard power supply with e.g. 5 V (higher than the value of "DigitalInputThreshold") to pin 1 (-) and pin 6 (+)

As long as the power supply is connected, you can see a live preview. If you disconnect the power supply the live preview should stop. If this is working the trigger input works.

Figure 40: Settings to test the general trigger functionality
External flash signal
For supplying flash with control signal use any digital output Out0 (pin 7 and 2) or Out1 (pin 8 and 3) on 9 pin D-Sub connector.

Flash current $\leq 100$ mA:

![Diagram of flash signal for $\leq 100$ mA current]

Flash current $> 100$ mA:

![Diagram of flash signal for $> 100$ mA current]

Figure 41: External flash signal

If current needed for flash is below 100 mA you can connect flash directly to camera outputs. If it is higher, you have to use an additional driver for controlling the flash, which provides the higher current.

Note
Depending on used flash driver it could be necessary to use pull-up or pull-down resistors so that driver can recognize the signal correctly.

See also
Characteristics of the digital outputs (p. 46)

11.1.2.5.2 Setting up

In wxPropView (p. 78) you can open camera and display acquired images.

By default camera is running free. This means, it uses its own timing depending on set pixel clock, exposure time and shutter mode.

Trigger
To let the camera acquire images only with an external trigger signal you must change the "TriggerMode" to the mode suitable to your application:
Figure 42: TriggerMode

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous</td>
<td>Free running, no external trigger signal needed.</td>
</tr>
<tr>
<td>OnDemand</td>
<td>Image acquisition triggered by command (software trigger).</td>
</tr>
<tr>
<td>OnLowLevel</td>
<td>As long as trigger signal is Low camera acquires images with own timing.</td>
</tr>
<tr>
<td>OnHighLevel</td>
<td>As long as trigger signal is High camera acquires images with own timing.</td>
</tr>
<tr>
<td>OnFallingEdge</td>
<td>Each falling edge of trigger signal acquires one image.</td>
</tr>
<tr>
<td>OnRisingEdge</td>
<td>Each rising edge of trigger signal acquires one image.</td>
</tr>
<tr>
<td>OnHighExpose</td>
<td>Each rising edge of trigger signal acquires one image, exposure time corresponds to pulse width.</td>
</tr>
<tr>
<td>OnLowExpose</td>
<td>Each falling edge of trigger signal acquires one image, exposure time corresponds to pulse width.</td>
</tr>
<tr>
<td>OnAnyEdge</td>
<td>Start the exposure of a frame when the trigger input level changes from high to low or from low to high.</td>
</tr>
</tbody>
</table>

Now, define on which pin the trigger signal is connected to.
Choose either "DigIn0" if signal is connected to "IN0" or DigIn1 if signal is connected to IN1. In general entry "RTCtrl" is not useful in this case because triggering would be controlled by Hardware Real-Time Controller (p.125) which is not described here and also not necessary.

Depending on voltage level you are using for trigger signal you must choose the "DigitalInputThreshold":

![DigitalInputThreshold Diagram]
In case of TTL choose 2V and in case of PLC choose 10V.

Now, the image acquisition runs corresponding to external trigger signal and trigger mode. You will see the acquired images on the left part of the window. Preview will be updated in frequency of external trigger.

The program knows a timeout period within at least one trigger signal must be provided to the camera. If no trigger signal comes within this time, no image is acquired and an error is set (error count increases). So be sure to set this timeout period to a value, which is long enough to receive at least one trigger signal. You can set this value in "ImageRequestTimeout" property:

![Figure 45: ImageRequestTimeout](image1.png)

**Trigger**

To activate flash control signal set "FlashMode" to the output you connected flash or flash driver to:

![Figure 46: External flash signal](image2.png)
Since this mode is activated each image acquisition will generate the flash signal. This generation is independent from used trigger mode. Flash signal is directly derived from integration timing.

This means that if no "FlashToExposedToLightDelay" is set flash signal will rise as soon as integration starts and fall when integration is finished. The pulse width cannot be changed. So you can be sure that integration takes place when trigger signal is high.

11.1.2.6 Working with the hardware Look-Up-Table (LUT)

There are two parameters which handles the pixel formats of the camera:

- "Setting -> Camera -> PixelFormat" defines the pixel format used to transfer the image data into the target systems host memory.

- "Setting -> ImageDestination -> PixelFormat" defines the pixel format of the resulting image (which is kept in the memory by the driver).

If both formats are set to "Auto", 8 bit will be used.

If you set "LUTImplementation" to "Software" in "Setting -> ImageProcessing -> LUTOperations", the hardware Look-Up-Table (LUT) will work with 8 bit data ("LUTMappingSoftware = 8To8"). Using Gamma functions you will see gaps in the histogram:

![Figure 47: 8to8 software LUT leads to gaps in the histogram using gamma functions (screenshot: mvBlueFOX-MLC)](image-url)
If you set “LUTImplementation” to “Hardware” in “Setting -> ImageProcessing -> LUTOperations”, the hardware Look-Up-Table (LUT) will work with 10 bit data inside the camera and converts the data to 8 bit for output (“LUT← MappingHardware = 10To8”). Now, there will be no gaps in the histogram:

![Image showing histogram with no gaps](screenshot: mvBlueFOX-MLC)

Figure 48: 10to8 hardware LUT shows no gaps in the histogram (screenshot: mvBlueFOX-MLC)

11.1.3 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>
</tbody>
</table>
11.2 mvDeviceConfigure

mvDeviceConfigure (p. 117) 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.

11.2.1 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 tool. You will see the following window:

```
fulltree or ft
Will display the complete property tree (including the data not meant to be accessed by the user) in the property grid. Example (Tree will be shown):
:fulltree=1

device or d
Will directly open a device with a particular serial number.  will take the first device. Example: d=GX000735

qsw
Will forcefully hide or show the Quick Setup Wizard, regardless of the default settings. Example (Quick Setup Wizard will be shown): qsw=1

live
Will directly start live acquisition from the device opened via device or d directly. Example (will start the live acquisition): live=1

11.1.3.1 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.
```
Whenever there is a device that shares its ID with at least one other device belonging to the same device family, \texttt{mvDeviceConfigure} (p.117) will display a warning like in the following image, showing in this example two \texttt{mvBlueFOX} cameras with an ID conflict:

![Figure 47: mvDeviceConfigure - Overview devices](image1.png)

11.2.1.1 Step 1: Device Selection

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

11.2.1.2 Step 2: Open dialog to set the ID

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

\textbf{Note}

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

![Figure 48: mvDeviceConfigure - Conflicting device IDs](image2.png)

![Figure 49: mvDeviceConfigure - Select action](image3.png)
11.2.1.3 Step 3: Assign the new ID

Enter the new ID and click OK.

![Figure 50: mvDeviceConfigure - New ID](image)

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. 117) will no longer highlight the conflict now:

![Figure 51: mvDeviceConfigure - Resolved ID conflict](image)

11.2.2 How to update the firmware

With the mvDeviceConfigure tool it is also possible to update the firmware. These steps are necessary:

11.2.2.1 Step 1: Device selection

Select the device you want to update from the list box.
11.2.2.2 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 52: mvDeviceConfigure - Select action](image)

11.2.2.3 Step 3: Confirm the firmware update

You have to confirm the update.

![Figure 53: mvDeviceConfigure - Confirm update](image)

**Note**

The firmware is compiled within the installed driver. The *mvDeviceConfigure* uses this version and updates the firmware. If you use an old driver, you will downgrade the firmware.

If the firmware update is successful, you will receive the following message:

![Figure 54: mvDeviceConfigure - Update successful](image)
11.2.2.4 Step 4: Disconnect and reconnect the device

Please disconnect and reconnect the device to activate the new firmware.

**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). The firmware update takes approx. 30 seconds!

11.2.3 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`:

**Note**

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".
The sleep states can also be enabled or disabled from a script by calling `mvDeviceConfigure` like this:

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

or

```
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.

11.2.4 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:
11.2 mvDeviceConfigure

<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: `id=serial</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: `spis=1` or `spis=0`).</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: `sup=1` or `sup=0`).</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><code>*</code></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.

11.2.4.1 Sample (Windows)

mvDeviceConfigure ufw=BF000666

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

mvDeviceConfigure update_fw=BF*

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

mvDeviceConfigure update_fw=mvBlueFOX-2* 1f=output.txt quit

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.

mvDeviceConfigure setid=BF000666.5

MATRIX VISION GmbH
This will assign the device ID ‘5’ to a mvBlueFOX with the serial number BF000666.

```
mvDeviceConfigure ufw=* 
```

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

```
mvDeviceConfigure ufw=BF000666 ufw=BF000667 
```

This will update the firmware of 2 mvBlueFOX cameras.

```
mvDeviceConfigure ipv4_mask=169.254.*;192.168.100* update_fw=GX* 
```

This will update the firmware of all mvBlueCOUGAR-X devices with a valid IPv4 address that starts with ‘169.254.’ or ‘192.168.100.’.
**12 HRTC - Hardware Real-Time Controller**

**12.1 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 build into the camera. This is a very unique and powerful feature.

**12.1.1 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.

The section *How to use the HRTC* (p. 125) should give the impression what everything can be done with the HRTC.

**12.2 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
CameraSettings->triggerMode = ctmOnRisingEdge
CameraSettings->triggerSource = ctsRTCtrl
```

When working with *wxPropView* (p. 78) this are the properties to modify in order to activate the evaluation of the HRTC program:
Following trigger modes can be used with HRTC:

- OnLowLevel
- OnHighLevel
- OnFallingEdge
- OnRisingEdge
- OnHighExpose

Further details about the mode are described in the API documentation:

See also

TCameraTriggerMode and TCameraTriggerSource in

- "Enumerations (C developers)"
- "CameraSettingsBlueFOX (C++ developers)"

In the Use cases (p.140) chapter there are the following HRTC sample:

- "Using single camera" :
  - Achieve a defined image frequency (HRTC) (p.182)
  - Delay the external trigger signal (HRTC) (p.183)
  - Creating double acquisitions (HRTC) (p.185)
  - Take two images after one external trigger (HRTC) (p.185)
  - Take two images with different expose times after an external trigger (HRTC) (p.186)
- "Using multiple cameras" :
  - Delay the expose start of the following camera (HRTC) (p.190)
13 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.
14 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.
15 .NET developers

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.
16 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!

16.1 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.

16.2 Building

During the compilation process Pythons distutils package will be used

16.2.1 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 your 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.
16.3 Using

16.2.2 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!

16.3 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(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(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."

    # 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.isOK():
                    if i == 0:
                        print("Info from " + pDev.serial.read() + ", " + statistics.framesPerSecond.name() + ": " + statistics.framesPerSecond.readS() + ", " + statistics.errorCount.name() + ": " + statistics.errorCount.readS() + ", " + statistics.captureTime_s.name() + ": " + statistics.captureTime_s.readS())
                    if isDisplayModuleAvailable:
                        display.GetImageDisplay().SetImage(pRequest)
                        display.GetImageDisplay().Update()
                    # For systems with NO mvDisplay library support
                    cbuf = (ctypes.c_char * imageSize).from_address(long(req.imageData.read()))
                    arr = numpy.fromstring(cbuf, dtype = channelType)
                    arr.shape = (height, width, channelCount)
                    if channelCount == 1:
                        img = Image.fromarray(arr)
                    else:
                        img = Image.fromarray(arr, 'RGBA' if alpha else 'RGB')
                    if pPreviousRequest != None:
                        pPreviousRequest.unlock()
pPreviousRequest = pRequest
fi.imageRequestSingle()
else:
    print("imageRequestWaitFor failed (" + str(requestNr) + ", " + ImpactAcquireException.getErrorAsString(requestNr) + ")")
    manuallyStopAcquisitionIf Needed(pDev, fi)
    raw_input("Press Enter to continue...")

Note

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

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!
17 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. 134)
- Logging (p. 134)
- Registering and renaming devices for DirectShow usage (p. 135)

17.1 Supported Interfaces

17.1.1 IAMCameraControl

17.1.2 IAMDroppedFrames

17.1.3 IAMStreamConfig

17.1.4 IAMVideoProcAmp

17.1.5 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

17.1.6 ISpecifyPropertyPages

17.2 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.
17.3 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. 117). 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. 117) (`C:\Program Files\MATRIX VISION\mvI→MPACT Acquire\bin`)!

17.3.1 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)

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.
17.3 Registering and renaming devices for DirectShow usage

17.3.2 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":

   ![Figure 4: mvDeviceConfigure - set DirectShow friendly name]

3. Then, a dialog will appear. Please enter the new name and confirm it with "OK".

   ![New friendly name dialog]

MATRIX VISION GmbH
4. Afterwards the column "DirectShow friendly name" will display the newly assigned friendly name.

**Figure 6: mvDeviceConfigure - renamed device**

**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.

17.3.3 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
```
18 Troubleshooting

18.1 Accessing log files

If you need support using our products, you can shorten response times by sending us your log files. Accessing the log files is different in Windows and Linux:

18.1.1 Windows

Since mvIMPACT Acquire driver version 2.11.9 you can access the log files in Windows using *wxPropView* (p. 78). The way to do this is described in *Accessing log files* (p. 103).

18.1.2 Linux

Like in Windows, log files will be generated, if the activation flag for loggings called *mvDebugFlags.mvd* is available in the same folder as the application (however, using Windows log files will be generated automatically, because the applications are started from the same folder). By default, in Linux the *mvDebugFlags.mvd* will be installed in the installation's destination folder in the sub-folder "apps". For example, if the destination folder was "/home/workspace", you can locate the *mvDebugFlags.mvd* like the following way:

```
user@linux-desktop:~$ // <- Starting the console window
user@linux-desktop:~$ cd workspace/apps/ // <- Change the directory
user@linux-desktop:/home/workspace/apps$ ls -l // <- List the directory
insgesamt 144
drwxr-xr-x 9 user user 4096 Mai 21 15:08 Callback
drwxr-xr-x 8 user user 4096 Mai 21 15:08 Callback_C
drwxr-xr-x 9 user user 4096 Mai 21 15:08 CaptureToUserMemory_C
drwxr-xr-x 3 user user 4096 Mai 21 15:03 Common
drwxr-xr-x 9 user user 4096 Mai 21 15:09 ContinuousCaptureAllDevices
drwxr-xr-x 9 user user 4096 Mai 21 15:09 ContinuousCapture_C
drwxr-xr-x 11 user user 4096 Mai 21 15:09 DigitalI0s
drwxr-xr-x 9 user user 4096 Mai 21 15:09 FirmwareUpgrade
drwxr-xr-x 11 user user 4096 Mai 21 15:09 GenericInterfaceLayout
drwxr-xr-x 11 user user 4096 Mai 21 15:09 GenICamInterfaceLayout
drwxr-xr-x 11 user user 4096 Mai 21 15:09 LiveSnap
drwxr-xr-x 6 user user 4096 Mai 21 15:09 LiveSnapFLTK
-rw-r--r-- 1 user user 854 Mai 21 15:03 Makefile
-rw-r--r-- 1 user user 7365 Mai 21 15:03 Makefile.samp.inc
-rwxr-xr-x 1 user user 7186 Mai 21 15:03 mknewappl.sh
-rw-r--r-- 1 user user 20713 Mai 21 15:03 mvDebugFlags.mvd // <- Log activation flag
drwxr-xr-x 7 user user 4096 Mai 21 15:09 mvDeviceConfigure
drwxr-xr-x 6 user user 4096 Mai 21 15:10 mvIPConfigure
drwxr-xr-x 6 user user 4096 Mai 21 15:11 mvPropView
drwxr-xr-x 9 user user 4096 Mai 21 15:11 SingleCapture
drwxr-xr-x 9 user user 4096 Mai 21 15:11 SingleCaptureStorage
```

In order that log files are generated, you have to execute your app from the folder, where *mvDebugFlags.mvd* is located. E.g. if you want to start wxPropView:

```
user@linux-desktop:/home/workspace/apps$ ./mvPropView/x86/wxPropView // <- Start the executable from
```

Another possibility would be, to copy the *mvDebugFlags.mvd* file to the folder of the executable:

```
user@linux-desktop:/home/workspace/apps$ cp mvDebugFlags.mvd ./mvPropView/x86/wxPropView // <- Copy the log activation file
user@linux-desktop:/home/workspace/apps$ cd ./mvPropView/x86/
user@linux-desktop:/home/workspace/apps/mvPropView/x86/$ ./wxPropView // <- Start the executable
```

Afterwards, several log files are generated which are listed in *files.mvloglist*. The log files have the file extension .mvlog. Please send these files to our support team.
19 Use cases

19.1 Introducing acquisition / recording possibilities

There are several use cases concerning the acquisition / recording possibilities of the camera:

19.1.1 Generating very long exposure times

Since **mvIMPACT Acquire 1.10.65**

Very long exposure times are possible with mvBlueFOX. For this purpose a special trigger/IO mode is used. You can do this as follows (pseudo code):

```c
TriggerMode = OnHighExposure
TriggerSource = DigOUT0 - DigOUT3
```

**Attention**

In the standard mvBlueFOX DigOUT2 and DigOUT3 are internal signals, however, they can be used for this intention.

**Note**

Make sure that you adjust the `ImageRequestTimeout_ms` either to 0 (infinite)(this is the default value) or to a reasonable value that is larger than the actual exposure time in order not to end up with timeouts resulting from the buffer timeout being smaller than the actual time needed for exposing, transferring and capturing the image:

```
ImageRequestTimeout_ms = 0 # or reasonable value
```

Now request a single image:

```
imageRequestSingle
```

Then the digital output is set and reset. Between these two instructions you can include source code to get the desired exposure time.

```
# The DigOUT which was chosen in TriggerSource
DigitalOutput* pOut = getOutput(digital output)
pOut->set();

# Wait as long as the exposure should continue.
pOut->reset();
```

Afterwards you will get the image.

If you change the state of corresponding output twice this will also work with `wxPropView` (p. 78).
19.1.2 Using VLC Media Player

With the **DirectShow Interface** (p. 134) MATRIX VISION devices become a (acquisition) video device for the VLC Media Player.

![Figure 1: VLC Media Player with a connected device via DirectShow](image)

**19.1.2.1 System requirements**

It is necessary that following drivers and programs are installed on the host device (laptop or PC):

- Windows 7, 32 bit or 64 bit
- up-do-date VLC Media Player, 32 bit or 64 bit (here: version 2.0.6)
- up-do-date MATRIX VISION driver, 32 bit or 64 bit (here: version 2.5.6)

**Attention**

Using **Windows 10**: VLC Media Player with versions 2.2.0 or smaller have been tested successfully. Newer versions do **NOT** work with mvIMPACT Acquire! There are some bug tickets in the VLC repository that might be related (At the time of writing none of these seem to have a fix):

- [https://trac.videolan.org/vlc/ticket/14278](https://trac.videolan.org/vlc/ticket/14278) (According to [http://www.digital-digest.com/software/videolan_history.html#83](http://www.digital-digest.com/software/videolan_history.html#83) may have been fixed but the trac system suggests otherwise)
- [https://trac.videolan.org/vlc/ticket/14489](https://trac.videolan.org/vlc/ticket/14489) (According to [http://www.digital-digest.com/software/videolan_history.html#83](http://www.digital-digest.com/software/videolan_history.html#83) may have been fixed but the trac system suggests otherwise)
- [https://trac.videolan.org/vlc/ticket/15511](https://trac.videolan.org/vlc/ticket/15511)
19.1.2.2 Installing VLC Media Player

1. Download a suitable version of the VLC Media Player from the VLC Media Player website mentioned below.
2. Run the setup.
3. Follow the installation process and use the default settings.

A restart of the system is not required.

See also

http://www.videolan.org/

19.1.2.3 Setting up MV device for DirectShow

**Note**

Please be sure to register the MV device for DirectShow with the right version of `mvDeviceConfigure` (p. 117). I.e. if you have installed the 32 bit version of the VLC Media Player, you have to register the MV device with the 32 bit version of `mvDeviceConfigure` (p. 117) ("C:/Program Files/MATRIX VISION/mvIMPACT Acquire/bin") !

1. Connect the MV device to the host device directly or via GigE switch using an Ethernet cable.
2. Power the camera using a power supply at the power connector.
3. Wait until the status LED turns blue.
4. Open the tool `mvDeviceConfigure` (p. 117),
5. set a friendly name (p. 137),
6. and register the MV device for DirectShow (p. 135).

**Note**

In some cases it could be necessary to repeat step 5.

19.1.2.4 Working with VLC Media Player

1. Start VLC Media Player.
2. Click on "Media -> Open Capture Device...".
3. Select the tab "Device Selection".

4. In the section "Video device name", select the friendly name of the MV device:

![Video device name screenshot]

5. Finally, click on "Play".
   After a short delay you will see the live image of the camera.

19.2 Improving the acquisition / image quality

There are several use cases concerning the acquisition / image quality of the camera:
19.2.1 Correcting image errors of a sensor

Due to random process deviations, technical limitations of the sensors, etc. there are different reasons that image sensors have image errors. MATRIX VISION provides several procedures to correct these errors, by default these are host-based calculations.

Provided image corrections procedures are

1. **Defective Pixels Correction** (p. 145),
2. **Dark Current Correction** (p. 148), and
3. **Flat-Field Correction** (p. 150).

**Note**

If you execute all correction procedures, you have to keep this order. All gray value settings of the corrections below assume an 8-bit image.

![Host-based image corrections](image.png)

The path "Setting -> Base -> ImageProcessing -> ..." indicates that these corrections are host-based corrections.

Before starting consider the following hints:

- To correct the complete image, you have to make sure no user defined AOI has been selected: Right-click "Restore Default" on the devices AOI parameters \( W \) and \( H \) in "Setting -> Base -> Camera".
- You have several options to save the correction data. The chapter **Storing and restoring settings** (p. 87) describes the different ways.

**See also**

19.2.1.1 Defective Pixels Correction

Due to random process deviations, not all pixels in an image sensor array will react in the same way to a given light condition. These variations are known as blemishes or defective pixels.

There are three types of defective pixels:

1. **leaky pixel** (in the dark)
   which indicates pixels that produce a higher read out code than the average
2. **hot pixel** (in standard light conditions)
   which indicates pixels that produce a higher non-proportional read out code when temperatures are rising
3. **cold pixel** (in standard light conditions)
   which indicates pixels that produce a lower read out code than average when the sensor is exposed (e.g. caused by dust particles on the sensor)

**Note**

Please use either an 8 bit Mono or Bayer image format when correcting the image. After the correction, all image formats will be corrected.

To correct the defective pixels there are three neighbor replace methods:

1. **"Replace 3x1 average"**
   which replaces the detected defective pixels with the average value from the left and right neighbor pixel (3x1)
2. **"Replace 3x3 median"**
   which replaces the detected defective pixels with the median value calculated from the nearest neighbors (3x3)
3. **"Replace 3x3 Filtered Data Averaged"**
   which replaces and treats the detected defective pixels as being fed into a 3x3 debayer algorithm before reaching the filter
   Only for devices which do not offer a defective pixel compensation; packed RGB or packed YUV444 data is needed.

19.2.1.1.1 Correcting leaky pixels

To correct leaky pixels the following steps are necessary:

1. Set gain ("Setting -> Base -> Camera -> GenICam -> Analog Control -> Gain = 0 dB") and exposure time
   "Setting -> Base -> Camera -> GenICam -> Acquisition Control -> ExposureTime = 360 msec" to the given operating conditions
   The total number of defective pixels found in the array depend on the gain and the exposure time.
2. Black out the lens completely
3. Set the (Filter-) "Mode = Calibrate leaky pixel"
4. Snap an image ("Acquire" with "Acquisition Mode = SingleFrame")
5. To activate the correction, choose one of the neighbor replace methods: "Replace 3x1 average", "Replace 3x3 median" or "Replace 3x3 Filtered Data Averaged"
6. Save the settings including the correction data via "Action -> Capture Settings -> Save Active Device Settings"
   (Settings can be saved in the Windows registry or in a file)
Note

After having re-started the camera you have to reload the capture settings vice versa.

The filter checks:

Pixel > LeakyPixelDeviation_ADCLimit // (default value: 50)

All pixels above this value are considered as leaky pixel.

19.2.1.2 Correcting hot pixels

To correct **hot pixels** the following steps are necessary:

1. Set gain (**Setting -> Base -> Camera -> GenICam -> Analog Control -> Gain = 0 dB**) and exposure time
   (**Setting -> Base -> Camera -> GenICam -> Acquisition Control -> ExposureTime = 360 msec**) to the given
   operating conditions.
   The total number of defective pixels found in the array depend on the gain and the exposure time.

2. Black out the lens completely.

3. Set the (Filter-) **Mode = Calibrate Hot Pixel**.

4. Snap an image (**Acquire** with **Acquisition Mode = SingleFrame**).

5. To activate the correction, choose one of the neighbor replace methods: **Replace 3x1 average**, **Replace 3x3 median** or **Replace 3x3 Filtered Data Averaged**.

6. Save the settings including the correction data via **Action -> Capture Settings -> Save Active Device Settings**.
   (Settings can be saved in the Windows registry or in a file)

Note

After having re-started the camera you have to reload the capture settings vice versa.

19.2.1.3 Correcting cold pixels

To correct **cold pixels** the following steps are necessary:

1. You will need a uniform sensor illumination approx. 50 - 70 % saturation (which means an average gray value
   between 128 and 180).

2. Set the (Filter-) **Mode = Calibrate cold pixel** (Figure 2).

3. Snap an image (**Acquire** with **Acquisition Mode = SingleFrame**).

4. To activate the correction, choose one of the neighbor replace methods: **Replace 3x1 average**, **Replace 3x3 median** or **Replace 3x3 Filtered Data Averaged**.

5. Save the settings including the correction data via **Action -> Capture Settings -> Save Active Device Settings**.
   (Settings can be saved in the Windows registry or in a file)
Note

After having re-started the camera you have to reload the capture settings vice versa.

The filter checks:

\[
\text{Pixel} < T[\text{cold}] \quad \text{(default value: 15 \%)}
\]

// T[\text{cold}] = deviation of the average gray value (ColdPixelDeviation_pc)

All pixels below this value have a dynamic below normal behavior.

![DefectivePixelsFilter](image.png)

Figure 2: Image corrections: DefectivePixelsFilter

Note

Repeating the defective pixel corrections will accumulate the correction data which leads to a higher value in "DefectivePixelsFound". If you want to reset the correction data or repeat the correction process you have to set the (Filter-) "Mode = Reset Calibration Data".
19.2.1.2 Dark Current Correction

Dark current is a characteristic of image sensors, which means, that image sensors also deliver signals in total darkness by warmness, for example, which creates charge carriers spontaneously. This signal overlays the image information. Dark current depends on two circumstances:

1. **Exposure time**
   The longer the exposure, the greater the dark current part. I.e. using long exposure times, the dark current itself could lead to an overexposed sensor chip

2. **Temperature**
   By cooling the sensor chips the dark current production can be highly dropped (approx. every 6 °C the dark current is cut in half)

19.2.1.2.1 Correcting Dark Current

The dark current correction is a pixel wise correction where the dark current correction image removes the dark current from the original image. To get a better result it is necessary to snap the original and the dark current images with the same exposure time and temperature.

**Note**

Dark current snaps generally show noise.

To correct the **dark current pixels** following steps are necessary:

1. Black out the lens completely
2. Set "OffsetAutoCalibration = Off" (Figure 3)
3. If applicable, change Offset_pc until you'll see an amplitude in the histogram (Figure 4)
4. Set exposure time according to the application
5. Set the (Filter-) "Mode = Calibrate"
6. Snap an image ("Acquire" with "Acquisition Mode = SingleFrame")
7. Finally, you have to activate the correction: Set the (Filter-) "Mode = On"
8. Save the settings including the correction data via "Action -> Capture Settings -> Save Active Device Settings"
   (Settings can be saved in the Windows registry or in a file)

The filter snaps a number of images and averages the dark current images to one correction image.
### Note

After having re-started the camera you have to reload the capture settings vice versa.

![Image Corrections Screenshot](image)

Figure 3: Image corrections (screenshot mvBlueFOX): OffsetAutoCalibration = Off
19.2.1.3 Flat-Field Correction

Each pixel of a sensor chip is a single detector with its own properties. Particularly, this pertains to the sensitivity as the case may be the spectral sensitivity. To solve this problem (including lens and illumination variations), a plain and equally "colored" calibration plate (e.g. white or gray) as a flat-field is snapped, which will be used to correct the original image. Between flat-field correction and the future application you must not change the optic. To reduce errors while doing the flat-field correction, a saturation between 50 % and 75 % of the flat-field in the histogram is convenient.
Note

Flat-field correction can also be used as a destructive watermark and works for all f-stops.

To make a **flat field correction** following steps are necessary:

1. You need a plain and equally "colored" calibration plate (e.g. white or gray)

2. No single pixel may be saturated - that's why we recommend to set the maximum gray level in the brightest area to max. 75% of the gray scale (i.e., to gray values below 190 when using 8-bit values)

3. Choose a *BayerXY* in "Setting -> Base -> Camera -> GenICam -> Image Format Control -> PixelFormat".

4. Set the (Filter-) "Mode = Calibrate" (Figure 6)

5. Start a Live snap ("Acquire" with "Acquisition Mode = Continuous")

6. Finally, you have to activate the correction: Set the (Filter-) "Mode = On"

7. Save the settings including the correction data via "Action -> Capture Settings -> Save Active Device Settings"
   (Settings can be saved in the Windows registry or in a file)

Note

After having re-started the camera you have to reload the capture settings vice versa.

The filter snaps a number of images (according to the value of the *CalibrationImageCount*, e.g. 5) and averages the flat-field images to one correction image.
19.2.2 Optimizing the color fidelity of the camera

Purpose of this chapter is to optimize the color image of a camera, so that it looks as natural as possible on different displays and for human vision.

This implies some linear and nonlinear operations (e.g. display color space or Gamma viewing LUT) which are normally not necessary or recommended for machine vision algorithms. A standard monitor offers, for example, several display modes like sRGB, “Adobe RGB”, etc., which reproduce the very same color of a camera color differently.

It should also be noted that users can choose for either

- camera based settings and adjustments or
- host based settings and adjustments or

Figure 6: Image corrections: Host-based flat field correction
• a combination of both.

Camera based settings are advantageous to achieve highest calculating precision, independent of the transmission bit depth, lowest latency, because all calculations are performed in FPGA on the fly and low CPU load, because the host is not invoked with these tasks. These camera based settings are

• gamma correction (p. 155)
• negative gain / gain (p. 155)
• look-up table (LUT) (p. 155)
• white balance (p. 158)
• offset (p. 158)
• saturation and color correction (p. 159)

Host based settings save transmission bandwidth at the expense of accuracy or latency and CPU load. Especially performing gain, offset, and white balance in the camera while outputting RAW data to the host can be recommended.

Of course host based settings can be used with all families of cameras (e.g. also mvBlueFOX).

Host based settings are:

• look-up table (LUT)\textsc{Operations}
• color correction (\textsc{ColorTwist})

To show the different color behaviors, we take a color chart as a starting point:

![Color chart as a starting point](image)

Figure 1: Color chart as a starting point

If we take a \textit{SingleFrame} image without any color optimizations, an image can be like this:
As you can see,

- saturation is missing,
- white is more light gray,
- black is more dark gray,
- etc.

Note

You have to keep in mind that there are two types of images: the one generated in the camera and the other one displayed on the computer monitor. Up-to-date monitors offer different display modes with different color spaces (e.g. sRGB). According to the chosen color space, the display of the colors is different.

The following figure shows the way to a perfect colored image
19.2 Improving the acquisition / image quality

Figure 4: The way to a perfect colored image

including these process steps:

1. Do a Gamma correction (Luminance) (p. 155).
2. make a White balance (p. 158) and
3. Improve the Contrast (p. 158).
4. Improve Saturation (p. 159), and use a "color correction matrix" for both
   (a) the sensor and / or
   (b) the monitor.

The following sections will describe the single steps in detail.

19.2.2.1 Step 1: Gamma correction (Luminance)

First of all, a Gamma correction (Luminance) can be performed to change the image in a way how humans perceive light and color.

For this, you can change either

- the exposure time,
- the aperture or
- the gain.

You can change the gain via wxPropView (p. 78) like the following way:

1. Click on "Setting -> Base -> Camera". There you can find
   (a) "AutoGainControl" and
   (b) "AutoExposeControl".
You can turn them "On" or "Off". Using the auto controls you can set limits of the auto control; without you can set the exact value.

After gamma correction, the image will look like this:

![Figure 6: After gamma correction](image-url)
19.2 Improving the acquisition / image quality

Figure 7: Corresponding histogram after gamma correction

Note
As mentioned above, you can do a gamma correction via ("Setting -> Base -> ImageProcessing -> LUTOperations"):

Figure 8: LUTOperations dialog

Just set “LUTEnable” to “On” and adapt the single LUTs like (LUT-0, LUT-1, etc.).
19.2.2.2 Step 2: White Balance

As you can see in the histogram, the colors red and blue are below green. Using green as a reference, we can optimize the white balance via "Setting -> Base -> ImageProcessing" ("WhiteBalanceCalibration"):

Please have a look at White balance of a camera device (color version) (p.104) for more information for an automatic white balance.

To adapt the single colors you can use the "WhiteBalanceSettings-1".

After optimizing white balance, the image will look like this:

![Figure 9: After white balance](image)

![Figure 10: Corresponding histogram after white balance](image)

19.2.2.3 Step 3: Contrast

Still, black is more a darker gray. To optimize the contrast you can use "Setting -> Base -> ImageProcessing -> LUTControl" as shown in Figure 8.

The image will look like this now:
19.2 Improving the acquisition / image quality

19.2.2.4 Step 4: Saturation and Color Correction Matrix (CCM)

Still saturation is missing. To change this, the "Color Transformation Control" can be used ("Setting -> Base -> ImageProcessing -> ColorTwist"): 

1. Click on "Color Twist Enable" and

2. Click on "Wizard" to start the saturation via "Color Transformation Control" wizard tool (since firmware version 1.4.57).
3. Now, you can adjust the saturation e.g. "1.1".

4. Afterwards, click on "Enable".

5. Since driver version 2.2.2, it is possible to set the special color correction matrices at
   (a) the input (sensor),
   (b) the output side (monitor) and
   (c) the saturation itself using this wizard.

6. Select the specific input and output matrix and
7. click on “Enable”.

8. As you can see, the correction is done by the host ("Host Color Correction Controls").

**Note**

It is not possible to save the settings of the "Host Color Correction Controls" in the mvBlueFOX. Unlike in the case of Figure 14, the buttons to write the "Device Color Correction Controls" to the mvBlueFOX are not active.

9. Finally, click on “Apply”.

After the saturation, the image will look like this:

![Figure 15: After adapting saturation](image)

![Figure 16: Corresponding histogram after adapting saturation](image)

19.3 Working with triggers

There are several use cases concerning trigger:
19.3.1 Using external trigger with CMOS sensors

19.3.1.1 Scenario

The CMOS sensors used in mvBlueFOX cameras support the following trigger modes:

- Continuous
- OnDemand (software trigger)
- OnLowLevel
- OnHighLevel
- OnHighExpose (only with mvBlueFOX-[Model]205 (5.0 Mpix [2592 x 1944]) (p. 245))

If an external trigger signal occurs (e.g. high), the sensor will start to expose and readout one image. Now, if the trigger signal is still high, the sensor will start to expose and readout the next image (see Figure 1, upper part). This will lead to an acquisition just like using continuous trigger.

If you want to avoid this effect, you have to adjust the trigger signal. As you can see in Figure 1 (lower part), the possible period has to be smaller than the time an image will need ($t_{exposure} + t_{readout}$).

19.3.1.2 Example

19.3.1.2.1 External synchronized image acquisition (high active)
19.4 Working with HDR (High Dynamic Range Control)

There are several use cases concerning High Dynamic Range Control:

19.4.1 Adjusting sensor -x00w

19.4.1.1 Introduction

The HDR (High Dynamic Range) mode of the sensor -x00w increases the usable contrast range. This is achieved by dividing the integration time in two or three phases. The exposure time proportion of the three phases can be set independently. Furthermore, it can be set, how much signal of each phase is charged.

---

**Note**

Using mvBlueFOX-MLC or mvBlueFOX-IGC, you have to select DigIn0 as the trigger source, because the camera has only one opto-coupled input. Only the TTL model of the mvBlueFOX-MLC has two I/O's.

- **Trigger** modes
  
  - **OnHighLevel**:  
    The high level of the trigger has to be shorter than the frame time. In this case, the sensor will make one image exactly. If the high time is longer, there will be images with the possible frequency of the sensor as long as the high level takes. The first image will start with the low-high edge of the signal. The integration time of the exposure register will be used.
  
  - **OnLowLevel**:  
    The first image will start with the high-low edge of the signal.
  
  - **OnHighExposé**  
    This mode is like OnHighLevel, however, the exposure time is used like the high time of the signal.

See also

Block diagrams with example circuits of the opto-isolated digital inputs and outputs can be found in Dimensions and connectors (p. 60).
19.4.1.2 Functionality

![Diagram of the sensor's HDR mode](image)

Figure 1: Diagram of the sensor's HDR mode

19.4.1.2.1 Description

- **"Phase 0"**
  - During T1 all pixels are integrated until they reach the defined signal level of Knee Point 1.
  - If one pixel reaches the level, the integration will be stopped.
  - During T1 no pixel can reach a level higher than P1.

- **"Phase 1"**
  - During T2 all pixels are integrated until they reach the defined signal level of Knee Point 2.
  - T2 is always smaller than T1 so that the percentage compared to the total exposure time is lower.
  - Thus, the signal increase during T2 is lower as during T1.
  - The max. signal level of Knee Point 2 is higher than of Knee Point 1.

- **"Phase 2"**
  - During T3 all pixels are integrated until the possible saturation.
  - T3 is always smaller than T2, so that the percentage compared to the total exposure time is again lower here.
  - Thus, the signal increase during T3 is lower as during T2.

For this reason, darker pixels can be integrated during the complete integration time and the sensor reaches its full sensitivity. Pixels, which are limited at each Knee Points, lose a part of their integration time - even more, if they are brighter.
In the diagram you can see the signal line of three different bright pixels. The slope depends of the light intensity, thus it is per pixel the same here (granted that the light intensity is temporally constant). Given that the very light pixel is limited soon at the signal levels S1 and S2, the whole integration time is lower compared to the dark pixel. In practice, the parts of the integration time are very different. T1, for example, is 95\% of T_{total}, T2 only 4\% and T3 only 1\%. Thus, a high decrease of the very light pixels can be achieved. However, if you want to divide the integration thresholds into three parts that is S2 = 2 \times S1 and S3 = 3 \times S1, a hundredfold brightness of one pixel's step from S2 to S3, compared to the step from 0 and S1 is needed.

19.4.1.3 Using HDR with mvBlueFOX-x00w

Figure 3 is showing the usage of the HDR mode. Here, an image sequence was created with the integration time between 10us and 100ms. You can see three slopes of the HDR mode. The "waves" result from the rounding during the three exposure phases. They can only be partly adjusted during one line period of the sensor.
19.4.1.3.1 Notes about the usage of the HDR mode with mvBlueFOX-x00w

- In the HDR mode, the basic amplification is reduced by approx. 0.7, to utilize a huge, dynamic area of the sensor.
- If the manual gain is raised, this effect will be reversed.
- Exposure times, which are too low, make no sense. During the third phase, if the exposure time reaches a possible minimum (one line period), a sensible lower limit is reached.

19.4.1.3.2 Possible settings using mvBlueFOX-x00w

Possible settings of the mvBlueFOX-x00w in HDR mode are:

"HDREnable":

- "Off": Standard mode
- "On": HDR mode on, reduced amplification:

"HDRMode":

- "Fixed": Fixed setting with 2 Knee Points. modulation Phase 0 .. 33% / 1 .. 66% / 2 .. 100%
- "Fixed0": Phase 1 exposure 12.55% , Phase 2 31.25% of total exposure
- "Fixed1": Phase 1 exposure 6.25% , Phase 2 1.56% of total exposure
- "Fixed2": Phase 1 exposure 3.12% , Phase 2 0.78% of total exposure
- "Fixed3": Phase 1 exposure 1.56% , Phase 2 0.39% of total exposure
- "Fixed4": Phase 1 exposure 0.78% , Phase 2 0.195% of total exposure
- "Fixed5": Phase 1 exposure 0.39% , Phase 2 0.049% of total exposure

"User": Variable setting of the Knee Point (1..2), threshold and exposure time proportion

- "HDRKneePointCount": Number of Knee Points (1..2)
- "HDRKneePoints"
  - "HDRKneePoint-0"
    - "HDRExposure_ppm": Proportion of Phase 0 compared to total exposure in parts per million (ppm)
    - "HDRControlVoltage_mV": Control voltage for exposure threshold of first Knee Point (3030mV is equivalent to approx. 33%)
  - "HDRKneePoint-1"
    - "HDRExposure_ppm": Proportion of Phase 1 compared to total exposure in parts per million (ppm)
    - "HDRControlVoltage_mV": Control voltage for exposure threshold of first Knee Point (2630mV is equivalent to approx. 66%)
19.4.2 Adjusting sensor -x02d (-1012d)

19.4.2.1 Introduction

The HDR (High Dynamic Range) mode of the Aptina sensor increases the usable contrast range. This is achieved by dividing the integration time in three phases. The exposure time proportion of the three phases can be set independently.

19.4.2.2 Functionality

To exceed the typical dynamic range, images are captured at 3 exposure times with given ratios for different exposure times. The figure shows a multiple exposure capture using 3 different exposure times.

![Multiple exposure capture using 3 different exposure times](image)

**Figure 1: Multiple exposure capture using 3 different exposure times**

**Note**

The longest exposure time (T1) represents the Exposure_us parameter you can set in wxPropView.

Afterwards, the signal is fully linearized before going through a compander to be output as a piece-wise linear signal. The next figure shows this.
19.4.2.2.1 Description

Exposure ratios can be controlled by the program. Two ratios are used: \( R_1 = \frac{T_1}{T_2} \) and \( R_2 = \frac{T_2}{T_3} \).

Increasing \( R_1 \) and \( R_2 \) will increase the dynamic range of the sensor at the cost of lower signal-to-noise ratio (and vice versa).

19.4.2.2.2 Possible settings

Possible settings of the mvBlueFOX-x02d in HDR mode are:

- "HDREnable":
  - "Off": Standard mode
  - "On": HDR mode on, reduced amplification
- "HDRMode":
  - "Fixed": Fixed setting with exposure-time-ratios: \( T_1 \rightarrow T_2 \) ratio / \( T_2 \rightarrow T_3 \) ratio
  - "Fixed0": 8 / 4
  - "Fixed1": 4 / 8
  - "Fixed2": 8 / 8
  - "Fixed3": 8 / 16
  - "Fixed4": 16 / 16
  - "Fixed5": 16 / 32
### 19.5 Working with LUTs

There are several use cases concerning LUTs (Look-Up-Tables):

![Diagram](image.png)

*Figure 3: wxPropView - Working with the HDR mode*
19.5.1 Introducing LUTs

19.5.1.1 Introduction

Look-Up-Tables (LUT) are used to transform input data into a desirable output format. For example, if you want to invert an 8 bit image, a Look-Up-Table will look like the following:

![Look-Up-Table](image)

I.e., a pixel which is white in the input image (value 255) will become black (value 0) in the output image.

All MATRIX VISION devices use a hardware based LUT which means that

- no host CPU load is needed and
- the LUT operations are independent of the transmission bit depth.

19.5.1.2 Setting the hardware based LUTs via LUT Control

Note

The mvBlueFOX cameras also feature a hardware based LUT. Although, you have to set the LUT via Setting -> Base -> ImageProcessing -> LUTOperations (p. 170), you can set where the processing takes place. For this reason, there is the parameter LUTImplementation. Just select either "Software" or "Hardware".

19.5.1.3 Setting the Host based LUTs via LUTOperations

Host based LUTs are also available via "Setting -> Base -> ImageProcessing -> LUTOperations"). Here, the changes will affect the 8 bit image data and the processing needs the CPU of the host system.

The mvBlueFOX cameras also feature a hardware based LUT. Although, you have to set the LUT via "Setting -> Base -> ImageProcessing -> LUTOperations", you can set where the processing takes place. For this reason, there is the parameter LUTImplementation. Just select either "Software" or "Hardware".

Three "LUTMode"s are available:
• "Gamma"
  You can use "Gamma" to lift darker image areas and to flatten the brighter ones. This compensates the
  contrast of the object. The calculation is described here. It makes sense to set the "GammaStartThreshold"
  higher than 0 to avoid a too extreme lift or noise in the darker areas.

• "Interpolated"
  With "Interpolated" you can set the key points of a characteristic line. You can defined the number of key
  points. The following figure shows the behavior of all 3 LUTInterpolationModes with 3 key points:

  ![Interpolated](image)
  
  Figure 2: LUTMode "Interpolated" -> LUTInterpolationMode

• "Direct"
  With "Direct" you can set the LUT values directly.

19.5.1.3.1 Example 1: Inverting an Image

To get an inverted 8 bit mono image like shown in Figure 1, you can set the LUT using wxPropView (p. 78). After
starting wxPropView (p. 78) and using the device,

1. Set "LUTEnable" to "On" in "Setting -> Base -> ImageProcessing -> LUTOperations".
2. Afterwards, set "LUTMode" to "Direct".
3. Right-click on "LUTs -> LUT-0 -> DirectValues[256]" and select "Set Multiple Elements... -> Via A User
   Defined Value Range".
   This is one way to get an inverted result. It is also possible to use the "LUTMode" - "Interpolated".
4. Now you can set the range from 0 to 255 and the values from 255 to 0 as shown in Figure 2.
19.6 Saving data on the device

**Note**

As described in *Storing and restoring settings* (p. 87), it is also possible to save the settings as an XML file on the host system. You can find further information about for example the XML compatibilities of the different driver versions in the mvIMPACT Acquire SDK manuals and the according setting classes: [https://www.matrix-vision.com/manuals/SDK_CPP/classmvIMPACT_1_1acquire_1_1FunctionInterface.html](https://www.matrix-vision.com/manuals/SDK_CPP/classmvIMPACT_1_1acquire_1_1FunctionInterface.html)

There are several use cases concerning device memory:
19.6 Saving data on the device

19.6.1 Creating user data entries

19.6.1.1 Basics about user data

It is possible to save arbitrary user specific data on the hardware’s non-volatile memory. The amount of possible entries depends on the length of the individual entries as well as the size of the devices non-volatile memory reserved for storing:

- mvBlueFOX,
- mvBlueFOX-M,
- mvBlueFOX-MLC,
- mvBlueFOX3, and
- mvBlueCOUGAR-X

currently offer 512 bytes of user accessible non-volatile memory of which 12 bytes are needed to store header information leaving 500 bytes for user specific data.

One entry will currently consume:
1 + <length_of_name (up to 255 chars)> + 2 + <length_of_data (up to 65535 bytes)> + 1 (access mode) bytes

as well as an optional:
1 + <length_of_password> bytes per entry if a password has been defined for this particular entry

It is possible to save either String or Binary data in the data property of each entry. When storing binary data please note, that this data internally will be stored in Base64 format thus the amount of memory required is 4/3 time the binary data size.

The UserData can be accessed and created using wxPropView (p. 78) (the device has to be closed). In the section "UserData" you will find the entries and following methods:

- "CreateUserDataEntry"
- "DeleteUserDataEntry"
- "WriteDataToHardware"
To create a user data entry, you have to

- Right click on "CreateUserDataEntry"
- Select "Execute" from the popup menu.

An entry will be created.

- In "Entries" click on the entry you want to adjust and modify the data fields.

To permanently commit a modification made with the keyboard the ENTER key must be pressed.

- To save the data on the device, you have to execute "WriteDataToHardware". Please have a look at the "Output" tab in the lower right section of the screen as shown in Figure 2, to see if the write process returned with no errors. If an error occurs a message box will pop up.
19.6.1.2 Coding sample

If you e.g. want to use the UserData as dongle mechanism (with binary data), it is not suggested to use `wxPropView` (p. 78). In this case you have to program the handling of the user data.

See also

`mvIMPACT::acquire::UserDataEntry` in `mvIMPACT_Acquire_API_CPP_manual.chm`.

19.7 Working with device features

There are several use cases concerning device features:

19.8 Working with several cameras simultaneously

There are several use cases concerning multiple cameras:

19.8.1 Using 2 mvBlueFOX-MLC cameras in Master-Slave mode

19.8.1.1 Scenario

If you want to have a synchronized stereo camera array (e.g. mvBlueFOX-MLC-202dG) with a rolling shutter master camera (e.g. mvBlueFOX-MLC-202dC), you can solve this task as follows:

1. Please check, if all mvBlueFOX cameras are using firmware version 1.12.16 or newer.
2. Now, open `wxPropView` (p. 78) and set the master camera:

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlashMode</td>
<td>&quot;Digout0&quot;</td>
</tr>
<tr>
<td>FlashType</td>
<td>VSync</td>
</tr>
<tr>
<td>TestMode</td>
<td>Off</td>
</tr>
<tr>
<td>ShutterMode</td>
<td>ElectronicRollingShutter</td>
</tr>
<tr>
<td>ImageRequestTimeout_ms</td>
<td>2000 ms</td>
</tr>
<tr>
<td>FlashToExposeDelay_us</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 1: `wxPropView` - Master camera outputs at DigOut 0 a frame synchronous V-Sync pulse
Alternatively, it is also possible to use HRTC - Hardware Real-Time Controller (p. 125) HRTC to set the master camera. The following sample shows the HRTC - Hardware Real-Time Controller (p. 125) HRTC program which sets the trigger signal and the digital output. The sample will lead to a constant frame rate of 16 fps (50000 us + 10000 us = 60000 us for one cycle. $\frac{1}{60000 \text{ us}} + 100000 = 16.67 \text{ Hz}$).
Figure 2: wxPropView - HRTC program sets the trigger signal and the digital output

Do not forget to set HRTC as the trigger source for the master camera.
3. Then, set the slave with **wxPropView** (p. 78): 

![Figure 4: wxPropView - Slave camera with TriggerMode "OnHighLevel" at DigIn 0](image)

19.8.1.1 Connection using -UOW versions (opto-isolated inputs and outputs)

The connection of the mvBlueFOX cameras should be like this:
19.8 Working with several cameras simultaneously

Figure 5: Connection with opto-isolated digital inputs and outputs

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Comment</th>
<th>Input voltage</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>( U_{\text{ext.}} )</td>
<td>External power</td>
<td></td>
<td>3</td>
<td>30</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>( R_{\text{out}} )</td>
<td>Resistor digital output</td>
<td>3.3 V .. 5 V</td>
<td>2</td>
<td></td>
<td></td>
<td>kOhm</td>
</tr>
<tr>
<td>( R_{\text{in}} )</td>
<td>Resistor digital input</td>
<td>12 V</td>
<td>0.68</td>
<td></td>
<td></td>
<td>kOhm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>24 V</td>
<td>2</td>
<td></td>
<td></td>
<td>kOhm</td>
</tr>
</tbody>
</table>

You can add further slaves.

19.8.1.1.2 Connection using -UTW versions (TTL inputs and outputs)

The connection of the mvBlueFOX cameras should be like this:

Figure 6: Connection with TTL digital inputs and outputs

For this case we offer a synchronization cable called "KS-MLC-IO-TTL 00.5".
Note

There are no further slaves possible.

See also

- Dimensions and connectors (p. 60) Figure 18 pin reference.
- Dimensions and connectors (p. 60) Table of connector pinout of "12-pin through-hole type shrouded header (USB / Dig I/O)."
- Dimensions and connectors (p. 60) Electrical drawing "opto-isolated digital inputs" and "opto-isolated digital outputs".
- A predefined frame rate is also possible using HRTC. (p. 182)

19.8.2 Synchronize the cameras to expose at the same time

This can be achieved by connecting the same external trigger signal to one of the digital inputs of each camera like it’s shown in the following figure:

![Figure 1: Electrical setup for sync. cameras](image)

Each camera then has to be configured for external trigger somehow like in the image below:
This assumes that the image acquisition shall start with the rising edge of the trigger signal. Every camera must be configured like this. Each rising edge of the external trigger signal then will start the exposure of a new image at the same time on each camera. Every trigger signal that will occur during the exposure of an image will be silently discarded.

19.9 Working with the Hardware Real-Time Controller (HRTC)

**Note**

Please have a look at the **Hardware Real-Time Controller (HRTC)** (p. 125) chapter for basic information.

There are several use cases concerning the Hardware Real-Time Controller (HRTC):

- "Using single camera";
- "Using multiple cameras";
19.9.1 Single camera samples (HRTC)

Note
Please have a look at the Hardware Real-Time Controller (HRTC) chapter for basic information.

Using a single camera there are following samples available:

- Achieve a defined image frequency (HRTC)
- Delay the external trigger signal (HRTC)
- Take two images after one external trigger (HRTC)
- Take two images with different expose times after an external trigger (HRTC)
- Edge controlled triggering (HRTC)

19.9.1.1 Achieve a defined image frequency (HRTC)

Note
Please have a look at the Hardware Real-Time Controller (HRTC) chapter for basic information.

With the use of the HRTC, any feasible frequency with the accuracy of micro seconds(us) is possible. The program to achieve this roughly must look like this (with the trigger mode set to ctmOnRisingEdge):

0. WaitClocks( <frame time in us> - <trigger pulse width in us> )
1. TriggerSet 1
2. WaitClocks( <trigger pulse width in us> )
3. TriggerReset
4. Jump 0

So to get e.g. exactly 10 images per second from the camera the program would somehow look like this (of course the expose time then must be smaller or equal then the frame time in normal shutter mode):

0. WaitClocks 99000
1. TriggerSet 1
2. WaitClocks 1000
3. TriggerReset
4. Jump 0
19.9 Working with the Hardware Real-Time Controller (HRTC)

Figure 1: wxPropView - Entering the sample “Achieve a defined image frequency"

See also

Download this sample as an rtp file: Frequency10Hz.rtp. To open the file in wxPropView (p. 78), click on "Digital I/O -> HardwareRealTimeController -> Filename" and select the downloaded file. Afterwards, click on “int Load()” to load the HRTC program.

Note

Please note the max. frame rate of the corresponding sensor!

To see a code sample (in C++) how this can be implemented in an application see the description of the class mvIMPACT::acquire::RTCtrProgram (C++ developers)

19.9.1.2 Delay the external trigger signal (HRTC)
Please have a look at the Hardware Real-Time Controller (HRTC) (p. 125) chapter for basic information.

0. WaitDigIn DigIn0->On
1. WaitClocks <delay time>
2. TriggerSet 0
3. WaitClocks <trigger pulse width>
4. TriggerReset
5. Jump 0

<trigger pulse width> should not less than 100us.

![Program Step Diagram](image.png)

**Note**

WaitDigIn waits for a state.
Between TriggerSet and TriggerReset has to be a waiting period.
If you are waiting for an external edge in a HRTC sequence like

```
WaitDigIn[On,Ignore]
WaitDigIn[Off,Ignore]
```

the minimum pulse width which can be detected by HRTC has to be at least 5 us.

Figure 1: Delay the external trigger signal

As soon as digital input 0 changes from high to low (0), the HRTC waits the < delay time > (1) and starts the image expose. The expose time is used from the expose setting of the camera. Step (5) jumps back to the beginning to be able to wait for the next incoming signal.
19.9.1.3 Creating double acquisitions (HRTC)

**Note**

Please have a look at the Hardware Real-Time Controller (HRTC) chapter for basic information.

If you need a double acquisition, i.e. take two images in a very short time interval, you can achieve this by using the HRTC.

With the following HRTC code, you will

- take an image using `TriggerSet` and after `TriggerReset` you have to
- set the camera to `ExposureSet` immediately.
- Now, you have to wait until the first image was read-out and then
- set the second `TriggerSet`.

The ExposureTime was set to 200 us.

```
0  WaitDigIn DigitalInputs[0] - On
1  TriggerSet 1
2  WaitClocks 200
3  TriggerReset
4  WaitClocks 5
5  ExposureSet
6  WaitClocks 60000
7  TriggerSet 2
8  WaitClocks 100
9  TriggerReset
10 ExposureReset
11 WaitClocks 60000
12 Jump 0
```

19.9.1.4 Take two images after one external trigger (HRTC)

**Note**

Please have a look at the Hardware Real-Time Controller (HRTC) chapter for basic information.

```
0. WaitDigIn DigInD->Off
1. TriggerSet 1
2. WaitClocks <trigger pulse width>
3. TriggerReset
4. WaitClocks <time between 2 acquisitions - 10us> (= WC1)
5. TriggerSet 2
6. WaitClocks <trigger pulse width>
7. TriggerReset
8. Jump 0
```

<trigger pulse width> should not less than 100us.
This program generates two internal trigger signals after the digital input 0 is going to low. The time between those internal trigger signals is defined by step (4). Each image is getting a different frame ID. The first one has the number 1, defined in the command (1) and the second image will have the number 2. The application can ask for the frame ID of each image, so well known which image is the first and the second one.

19.9.1.5 Take two images with different expose times after an external trigger (HRTC)

Note Please have a look at the Hardware Real-Time Controller (HRTC) (p. 125) chapter for basic information.

The following code shows the solution in combination with a CCD model of the camera. With CCD models you have to set the exposure time using the trigger width.

0. WaitDigin DigIn0->Off
1. ExposeSet
2. WaitClocks <expose time image1 - 10us> (= WC1)
3. TriggerSet 1
4. WaitClocks <trigger pulse width>
5. TriggerReset
6. ExposeReset
7. WaitClocks <time between 2 acquisitions - expose time image1 - 10us> (= WC2)
8. ExposeSet
9. WaitClocks <expose time image2 - 10us> (= WC3)
10. TriggerSet 2
11. WaitClocks <trigger pulse width>
12. TriggerReset
13. ExposeReset
14. Jump 0

<trigger pulse width> should not less than 100us.
19.9 Working with the Hardware Real-Time Controller (HRTC) 187

Figure 1: Take two images with different expose times after an external trigger

Note

Due to the internal loop to wait for a trigger signal, the WaitClocks call between "TriggerSet 1" and "TriggerReset" constitute 100. For this reason, the trigger signal cannot be missed. Before the ExposeReset, you have to call the TriggerReset otherwise the normal flow will continue and the image data will be lost!

The sensor expose time after the TriggerSet is 0.

See also

Download this sample as an rtp file: 2Images2DifferentExposureTimes.rtp with two consecutive exposure times (10ms / 20ms). To open the file in wxPropView (p. 78), click on "Digital I/O -> HardwareRealTimeController -> Filename" and select the downloaded file. Afterwards, click on "int Load( )" to load the HRTC program. There are timeouts added in line 4 and line 14 to illustrate the different exposure times.

Using a CMOS model (e.g. the mvBlueFOX-MLC205), a sample with four consecutive exposure times (10ms / 20ms / 40ms / 80ms) triggered just by one hardware input signal would look like this:

0. WaitDigin DigIn0->On
1. TriggerSet
2. WaitClocks 10000 (= 10 ms)
3. TriggerReset
4. WaitClocks 100000 (= 1 s)
5. TriggerSet
6. WaitClocks 20000 (= 20 ms)
7. TriggerReset
8. WaitClocks 100000 (= 1 s)
9. TriggerSet
10. WaitClocks 40000 (= 40 ms)
11. TriggerReset
12. WaitClocks 100000 (= 1 s)
13. TriggerSet
14. WaitClocks 80000 (= 80 ms)
15. TriggerReset
16. WaitClocks 100000 (= 1 s)
17. Jump 0

See also

This second sample is also available as an rtp file: MLC205_four_images_diff_exp.rtp.
19.9.1.6 Edge controlled triggering (HRTC)

**Note**

Please have a look at the Hardware Real-Time Controller (HRTC) (p. 125) chapter for basic information.

To achieve an edged controlled triggering, you can use HRTC. Please follow these steps:

1. First of all, you have to set the **TriggerMode** to **OnHighLevel**.
2. Then, set the **TriggerSource** to **RTCtrl**.

<table>
<thead>
<tr>
<th>OffsetAutoBlackLevel</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>OffsetAutoBlackSpeed</td>
<td>Medium</td>
</tr>
<tr>
<td><strong>TriggerMode</strong></td>
<td><strong>OnHighLevel</strong></td>
</tr>
<tr>
<td><strong>TriggerSource</strong></td>
<td><strong>RTCtrl</strong></td>
</tr>
<tr>
<td>PixelClock_MHz</td>
<td>27000</td>
</tr>
<tr>
<td>Expose_us</td>
<td>2000</td>
</tr>
<tr>
<td>BirningMode</td>
<td>Off</td>
</tr>
<tr>
<td>FrameDelay_us</td>
<td>0</td>
</tr>
<tr>
<td>LineDelay_clk</td>
<td>0</td>
</tr>
<tr>
<td>FlashMode</td>
<td>Off</td>
</tr>
<tr>
<td>FlashType</td>
<td>Standard</td>
</tr>
<tr>
<td>TestMode</td>
<td>Off</td>
</tr>
<tr>
<td>ShutterMode</td>
<td>FrameShutter</td>
</tr>
<tr>
<td>ImageRequestTimeout_ms</td>
<td>0 ms</td>
</tr>
<tr>
<td>FlashToExposeDelay_us</td>
<td>0</td>
</tr>
<tr>
<td>AdvancedOptions</td>
<td>Off</td>
</tr>
</tbody>
</table>

Figure 1: wxPropView - TriggerMode and TriggerSource

Afterwards you have to configure the HRTC program:

1. The HRTC program waits for a rising edge at the digital input 0 (step 1).
2. If there is a rising edge, the trigger will be set (step 2).
3. After a short wait time (step 3),
4. the trigger will be reset (step 4).
5. Now, the HRTC program waits for a falling edge at the digital input 0 (step 5).
6. If there is a falling edge, the trigger will jump to step 0 (step 6).
19.9 Working with the Hardware Real-Time Controller (HRTC)

The waiting time at step 0 is necessary to debounce the signal level at the input (the duration should be shorter than the frame time).

<table>
<thead>
<tr>
<th>HardwareRealTimeController</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRTCctrl_0</td>
</tr>
<tr>
<td>ProgramSize</td>
</tr>
<tr>
<td>RTCtrProgram</td>
</tr>
<tr>
<td>Step0</td>
</tr>
<tr>
<td>OpCode</td>
</tr>
<tr>
<td>Clocks_us</td>
</tr>
<tr>
<td>Step1</td>
</tr>
<tr>
<td>OpCode</td>
</tr>
<tr>
<td>DigitalInputs[2]</td>
</tr>
<tr>
<td>DigitalInputs[0]</td>
</tr>
<tr>
<td>DigitalInputs[1]</td>
</tr>
<tr>
<td>Step2</td>
</tr>
<tr>
<td>OpCode</td>
</tr>
<tr>
<td>FrameID</td>
</tr>
<tr>
<td>Step3</td>
</tr>
<tr>
<td>OpCode</td>
</tr>
<tr>
<td>Clocks_us</td>
</tr>
<tr>
<td>Step4</td>
</tr>
<tr>
<td>OpCode</td>
</tr>
<tr>
<td>Step5</td>
</tr>
<tr>
<td>OpCode</td>
</tr>
<tr>
<td>DigitalInputs[2]</td>
</tr>
<tr>
<td>DigitalInputs[0]</td>
</tr>
<tr>
<td>DigitalInputs[1]</td>
</tr>
<tr>
<td>Step6</td>
</tr>
<tr>
<td>OpCode</td>
</tr>
<tr>
<td>Address</td>
</tr>
<tr>
<td>Mode</td>
</tr>
<tr>
<td>ProgramState</td>
</tr>
</tbody>
</table>

Figure 2: wxPropView - Edge controller triggering using HRTC

See also

Download this sample as a capture settings file: MLC200wG_HRTC_TriggerFromHighLevelToEdgeControl.xml. How you can work with capture settings is described in the following chapter (p. 87).

To see a code sample (in C++) how this can be implemented in an application see the description of the class mvIMPACT::acquire::RTCtrProgram (C++ developers)
19.9.2 Multiple camera samples (HRTC)

Note

Please have a look at the Hardware Real-Time Controller (HRTC) (p. 125) chapter for basic information.

Using a multiple camera there are following samples available:

- Delay the expose start of the following camera (HRTC) (p. 190)

19.9.2.1 Delay the expose start of the following camera (HRTC)

Note

Please have a look at the Hardware Real-Time Controller (HRTC) (p. 125) chapter for basic information. The use case Synchronize the cameras to expose at the same time (p. 180) shows how you have to connect the cameras.

If a defined delay should be necessary between the cameras, the HRTC can do the synchronization work.

In this case, one camera must be the master. The external trigger signal that will start the acquisition must be connected to one of the cameras digital inputs. One of the digital outputs then will be connected to the digital input of the next camera. So camera one uses its digital output to trigger camera two. How to connect the cameras to one another can also be seen in the following image:

Figure 1: Connection diagram for a defined delay from the exposure start of one camera relative to another

Assuming that the external trigger is connected to digital input 0 of camera one and digital output 0 is connected to digital input 0 of camera two. Each additional camera will then be connected to its predecessor like camera 2 is connected to camera 1. The HRTC of camera one then has to be programmed somehow like this:
0. WaitDigIn DigIn0->On
1. TriggerSet 0
2. WaitClocks <trigger pulse width>
3. TriggerReset
4. WaitClocks <delay time>
5. SetDigOut DigOut0->On
6. WaitClocks 100us
7. SetDigOut DigOut0->Off
8. Jump 0

<trigger pulse width> should not less than 100us.

When the cameras are set up to start the exposure on the rising edge of the signal <delay time> of course is the desired delay time minus <trigger pulse width>.

If more than two cameras shall be connected like this, every camera except the last one must run a program like the one discussed above. The delay times of course can vary.

![Diagram](image)

Figure 2: Delay the expose start of the following camera
20 Appendix A.1 CCD specific camera / sensor data

20.1 mvBlueFOX-[Model]220 (0.3 Mpix [640 x 480])

20.1.1 Introduction

The CCD sensor is a highly programmable imaging module which will, for example, enable the following type of applications

**Industrial applications:**

- triggered image acquisition with precise control of image exposure start by hardware trigger input.
- image acquisition of fast moving objects due to:
  - frame exposure, integrating all pixels at a time in contrast to CMOS imager which typically integrate line-by-line.
  - short shutter time, to get sharp images.
  - flash control output to have enough light for short time.

**Scientific applications:**

- long time exposure for low light conditions.
- optimizing image quality using the variable shutter control.

20.1.2 Details of operation

The process of getting an image from the CCD sensor can be separated into three different phases.

20.1.2.1 Trigger

When coming out of reset or ready with the last readout the CCD controller is waiting for a Trigger signal.

The following trigger modes are available:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous</td>
<td>Free running, no external trigger signal needed.</td>
</tr>
<tr>
<td>OnDemand</td>
<td>Image acquisition triggered by command (software trigger).</td>
</tr>
<tr>
<td>OnLowLevel</td>
<td>Start an exposure of a frame as long as the trigger input is below the trigger threshold.</td>
</tr>
<tr>
<td>OnHighLevel</td>
<td>Start an exposure of a frame as long as the trigger input is above the trigger threshold.</td>
</tr>
<tr>
<td>OnFallingEdge</td>
<td>Each falling edge of trigger signal acquires one image.</td>
</tr>
<tr>
<td>OnRisingEdge</td>
<td>Each rising edge of trigger signal acquires one image.</td>
</tr>
<tr>
<td>OnHighExposure</td>
<td>Each rising edge of trigger signal acquires one image, exposure time corresponds to pulse width.</td>
</tr>
</tbody>
</table>
20.1.2.2 Exposure aka Integration

After an active trigger, the exposure phase starts with a maximum jitter of $t_{\text{trig}}$. If flash illumination is enabled in software the flash output will be activated exactly while the sensor chip is integrating light. Exposure time is adjustable by software in increments of $t_{\text{readline}}$.

20.1.2.3 Readout

When exposure is finished, the image is transferred to hidden storage cells on the CCD. Image data is then shifted out line-by-line and transferred to memory. Shifting out non active lines takes $t_{\text{vshift}}$, while shifting out active lines will consume $t_{\text{readline}}$. The number of active pixels per line will not have any impact on readout speed.

20.1.3 CCD Timing

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Pixel clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{\text{trig}}$</td>
<td>Time from trigger (internal or external) to exposure start</td>
<td>10us 12 MHz 24 MHz</td>
</tr>
<tr>
<td>$t_{\text{trans}}$</td>
<td>Image transfer time (move image to readout cells in CCD)</td>
<td>64us 32us</td>
</tr>
<tr>
<td>$t_{\text{readline}}$</td>
<td>time needed to readout a line</td>
<td>64us 32us</td>
</tr>
<tr>
<td>$t_{\text{wait}}$</td>
<td>minimal time to next trigger</td>
<td>64us 32us 1.6us</td>
</tr>
<tr>
<td>$t_{\text{exposure}}$</td>
<td>Exposure time</td>
<td>2us 2 - 128s</td>
</tr>
<tr>
<td>$t_{\text{readout}}$</td>
<td>Image readout time (move image from readout cells to memory)</td>
<td>$t_{\text{readout}} = (\text{ActiveLines} \times t_{\text{readline}}) + (510 - \text{ActiveLines}) \times t_{\text{vshift}}$</td>
</tr>
</tbody>
</table>
Note

In partial scan mode (readout window ysize < 480 lines).

To calculate the maximum frames per second (FPS\textsubscript{max}) you will need following formula (ExposeMode: Standard):

\[
FPS\textsubscript{max} = \frac{1}{t\text{\_trig} + t\text{\_readout} + t\text{\_exposure} + t\text{\_trans} + t\text{\_wait}}
\]

(ExposeMode: Overlapped):

\[
\begin{align*}
& t\text{\_trig} + t\text{\_readout} + t\text{\_trans} + t\text{\_wait} < t\text{\_exposure}: \\
& FPS\textsubscript{max} = \frac{1}{t\text{\_integ}}
\end{align*}
\]

\[
\begin{align*}
& t\text{\_trig} + t\text{\_readout} + t\text{\_trans} + t\text{\_wait} > t\text{\_exposure}: \\
& FPS\textsubscript{max} = \frac{1}{t\text{\_trig} + t\text{\_readout} + t\text{\_trans} + t\text{\_wait}}
\end{align*}
\]

Example: Frame rate as function of lines & exposure time

Now, when we insert the values using exposure time of, for example, 65 us, 100 lines and 12MHz pixel clock (ExposeMode: Standard):

\[
FPS\textsubscript{max} = \frac{1}{10\text{ }\mu\text{s} + ((100 \times 64\text{ }\mu\text{s}) + ((510 - 100) \times 4.85\text{ }\mu\text{s}) + 3.15\text{ }\mu\text{s}) + 65\text{ }\mu\text{s} + 64\text{ }\mu\text{s} + 64\text{ }\mu\text{s}}
\]

\[
= 0.0001266704667806700868 \text{ } 1 / \mu\text{s} \approx 126.7
\]

Note

The calculator returns the max. frame rate supported by the sensor. Please keep in mind that it will depend on the interface and the used image format if this frame rate can be transferred.

See also

To find out how to achieve any defined freq. below or equal to the achievable max. freq., please have a look at Achieve a defined image frequency (HRTC) (p. 182).

20.1.4 Reprogramming CCD Timing

Reprogramming the CCD Controller will happen when the following changes occur

- Changing the exposure time
- Changing the capture window
- Changing Trigger Modes

Reprogram time consists of two phases

1. Time needed to send data to the CCD controller depending on what is changed
   - exposure: abt 2.3ms
   - window: abt 4.6ms
   - trigger mode: from 5.90ms, varies with oldmode/newmode combination

2. Time to initialize (erase) the CCD chip after reprogramming this is fixed, abt 4.5 ms

So for example when reprogramming the capture window you will need (average values)

\[
\begin{align*}
&t\text{\_regprog} = change\text{\_window} + init\text{\_ccd} \\
&t\text{\_regprog} = 5\text{ }\text{ms} + 4.5\text{ }\text{ms} \\
&t\text{\_regprog} = 9.5\text{ }\text{ms}
\end{align*}
\]
20.1.5 CCD Sensor Data

Device Structure

- Interline CCD image sensor
- Image size: Diagonal 4.5mm (Type 1/4)
- Number of effective pixels: 659 (H) x 494 (V) approx. 330K pixels
- Total number of pixels: 692 (H) x 504 (V) approx. 350K pixels
- Chip size: 4.60mm (H) x 3.97mm (V)
- Unit cell size: 5.6um (H) x 5.6um (V)
- Optical black:
  - Horizontal (H) direction: Front 2 pixels, rear 31 pixels
  - Vertical (V) direction: Front 8 pixels, rear 2 pixels
- Number of dummy bits: Horizontal 16 Vertical 5
- Substrate material: Silicon

20.1.5.1 Characteristics

These zone definitions apply to both the color and gray scale version of the sensor.

Zone Definition of Video Signal Shading
### Image Sensor Characteristics

\( (T_a = 25^\circ C) \)

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
<th>Measurement method</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>G sensitivity</td>
<td>Sg</td>
<td>460</td>
<td>590</td>
<td></td>
<td>mV</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Sensitivity comparison</td>
<td>R</td>
<td>0.4</td>
<td>0.55</td>
<td>0.7</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>0.3</td>
<td>0.45</td>
<td>0.6</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Saturation signal</td>
<td>Vsat</td>
<td>500</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>( T_a = 60^\circ C )</td>
</tr>
<tr>
<td>Smear</td>
<td>Sm</td>
<td>0.0008</td>
<td>0.0025</td>
<td>%</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Video signal shading</td>
<td>SHg</td>
<td>20</td>
<td></td>
<td></td>
<td>%</td>
<td>4</td>
<td>Zone 0 and I</td>
</tr>
<tr>
<td>Uniformity between video</td>
<td>( \Delta Srg )</td>
<td>8</td>
<td></td>
<td></td>
<td>%</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>signal channels</td>
<td>( \Delta Sbg )</td>
<td>8</td>
<td></td>
<td></td>
<td>%</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Dark signal</td>
<td>Vdt</td>
<td>4</td>
<td></td>
<td></td>
<td>mV</td>
<td>6</td>
<td>( T_a = 60^\circ C )</td>
</tr>
<tr>
<td>Dark signal shading</td>
<td>( \Delta Vdt )</td>
<td>1</td>
<td></td>
<td></td>
<td>mV</td>
<td>7</td>
<td>( T_a = 60^\circ C )</td>
</tr>
<tr>
<td>Line crawl G</td>
<td>Lcg</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Line crawl R</td>
<td>Lcr</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Line crawl B</td>
<td>Lcb</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Lag</td>
<td>Lag</td>
<td>0.5</td>
<td></td>
<td></td>
<td>%</td>
<td>9</td>
<td></td>
</tr>
</tbody>
</table>

### Spectral Sensitivity Characteristics

(excludes lens characteristics and light source characteristics)
### Image Sensor Characteristics

(Ta = 25°C)

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
<th>Measurement method</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>S</td>
<td>560</td>
<td>700</td>
<td></td>
<td>mV</td>
<td>1</td>
<td>1/30s accumulation mode</td>
</tr>
<tr>
<td>Saturation signal</td>
<td>Vsat</td>
<td>500</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>Ta = 60°C</td>
</tr>
<tr>
<td>Smear</td>
<td>Sm</td>
<td>0.0006</td>
<td>0.0025</td>
<td>%</td>
<td>3</td>
<td>1/30s accumulation mode</td>
<td></td>
</tr>
<tr>
<td>Video signal shading</td>
<td>SH</td>
<td>20</td>
<td></td>
<td>25</td>
<td>%</td>
<td>4</td>
<td>Zone 0 and 1</td>
</tr>
<tr>
<td>Dark signal</td>
<td>Vdt</td>
<td>4</td>
<td></td>
<td></td>
<td>mV</td>
<td>5</td>
<td>Ta = 60°C</td>
</tr>
<tr>
<td>Dark signal shading</td>
<td>ΔVdt</td>
<td>1</td>
<td></td>
<td></td>
<td>mV</td>
<td>6</td>
<td>Ta = 60°C</td>
</tr>
<tr>
<td>Lag</td>
<td>Lag</td>
<td>0.5</td>
<td></td>
<td></td>
<td>%</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

### Spectral Sensitivity Characteristics

(excludes lens characteristics and light source characteristics)
0 Measurement conditions

1) In the following measurements, the device drive conditions are at the typical values of the bias and clock voltage conditions.

2) In the following measurements, spot blemishes are excluded and, unless otherwise specified, the optical black level (OB) is used as the reference for the signal output, which is taken as the value measured at point [OB] of the measurement system.

0 Definition of standard imaging conditions

1) Standard imaging condition I:
   Use a pattern box (luminance: 706cd/m², color temperature of 3200K halogen source) as a subject. (pattern for evaluation is not applicable.) Use a testing standard lens with CM500S (f = 1.0mm) as an IR cut filter and image at F8. The luminous intensity to the sensor receiving surface at this point is defined as the standard sensitivity testing luminous intensity.

2) Standard imaging condition II:
   Image a light source (color temperature of 3200K) with a uniformity of brightness within ±2% at all angles. Use a testing standard lens with CM500S (f = 1.0mm) as an IR cut filter. The luminous intensity is adjusted to the value indicated in each testing item by the lens diaphragm.

1. Sensitivity
   Set to standard imaging condition I. After selecting the electronic shutter mode with a shutter speed of 1/100s, measure the signal output (Vs) at the center of the screen, and substitute the value into the following formula.

\[ S = \frac{Vs \times 100}{30} \text{ [mV]} \]

2. Saturation signal
   Set to standard imaging condition II. After adjusting the luminous intensity to 10 times the intensity with the average value of the signal output, 150mV, measure the minimum value of the signal output.

3. Smear
   Set to standard imaging condition II. With the lens diaphragm at F5.6 to F8, first adjust the luminous intensity to 500 times the intensity with the average value of the signal output, 150mV. Then after the readout clock is stopped and the charge drain is executed by the electronic shutter at the respective H blankings, measure the maximum value (Vsm [mV]) of the signal output and substitute the value into the following formula.

\[ Sm = \frac{Vsm}{150} \times \frac{500}{100} \times 100 \% \text{ (1/10V method conversion value)} \]

4. Video signal shading
   Set to standard imaging condition II. With the lens diaphragm at F5.6 to F8, adjust the luminous intensity so that the average value of the signal output is 150mV. Then measure the maximum (Vmax [mV]) and minimum (Vmin [mV]) values of the signal output and substitute the values into the following formula.

\[ SH = \frac{(Vmax - Vmin)}{150} \times 100 \% \]

5. Dark signal
   Measure the average value of the signal output (Vdt [mV]) with the device ambient temperature 80°C and the device in the light-obstructed state, using the horizontal idle transfer level as a reference.
6. Dark signal shading
After measuring 5, measure the maximum (V\text{dmax} [mV]) and minimum (V\text{dmin} [mV]) values of the dark signal output and substitute the values into the following formula.

\[ \Delta V_{\text{dt}} = V_{\text{dmax}} - V_{\text{dmin}} [\text{mV}] \]

7. Lag
Adjust the signal output value generated by strobe light to 150mV. After setting the strobe light so that it strobos with the following timing, measure the residual signal (V_{\text{lag}}). Substitute the value into the following formula.

\[ \text{Lag} = (V_{\text{lag}}/150) \times 100 \% \]

---

**Measurement System**

---

**Note** Adjust the amplifier gain so that the gain between [A] and [B] equals 1.

---

20.1.6 CCD Signal Processing

The CCD signal is processed with an analog front-end and digitized by an 12 bit analog-to-digital converter (A\rightarrow DC). The analog front-end contains a programmable gain amplifier which is variable from 0db (gain=0) to 30dB (gain=255).
The 8 most significant bits of the ADC are captured to the frame buffer. This will give the following transfer function (based on the 8 bit digital code): 

\[
\text{Digital code [lsb]} = \text{ccd_signal[V]} \times 256\text{[lsb/V]} \times \exp(\text{gain[bB]}/20)\]

\[\text{lsb} : \text{least significant bit (smallest digital code change)}\]

20.1.7 Device Feature And Property List

20.1.7.1 Gray scale version

20.1.7.2 Color version

20.2 mvBlueFOX-[Model]220a (0.3 Mpix [640 x 480])

20.2.1 Introduction

The CCD sensor is a highly programmable imaging module which will, for example, enable the following type of applications

**Industrial applications:**

- triggered image acquisition with precise control of image integration start by hardware trigger input.
- image acquisition of fast moving objects due to:
  - frame integration, integrating all pixels at a time in contrast to CMOS imager which typically integrate line-by-line.
  - short shutter time, to get sharp images.
  - flash control output to have enough light for short time.

**Scientific applications:**

- long time integration for low light conditions.
- optimizing image quality using the variable shutter control.

20.2.2 Details of operation

The process of getting an image from the CCD sensor can be separated into three different phases.

20.2.2.1 Trigger

When coming out of reset or ready with the last readout the CCD controller is waiting for a Trigger signal.

The following trigger modes are available:
<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous</td>
<td>Free running, no external trigger signal needed.</td>
</tr>
<tr>
<td>OnDemand</td>
<td>Image acquisition triggered by command (software trigger).</td>
</tr>
<tr>
<td>OnLowLevel</td>
<td>As long as trigger signal is Low camera acquires images with own timing.</td>
</tr>
<tr>
<td>OnHighLevel</td>
<td>As long as trigger signal is High camera acquires images with own timing.</td>
</tr>
<tr>
<td>OnFallingEdge</td>
<td>Each falling edge of trigger signal acquires one image.</td>
</tr>
<tr>
<td>OnRisingEdge</td>
<td>Each rising edge of trigger signal acquires one image.</td>
</tr>
<tr>
<td>OnHighExposure</td>
<td>Each rising edge of trigger signal acquires one image, exposure time corresponds to pulse width.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TriggerSource</th>
<th>mvIMPACT Acquire</th>
</tr>
</thead>
<tbody>
<tr>
<td>GP-IN0</td>
<td>GP-IN1</td>
</tr>
<tr>
<td>TriggerSource</td>
<td>GenICam(BCX)</td>
</tr>
<tr>
<td>Line4</td>
<td>Line5</td>
</tr>
</tbody>
</table>

See also
For detailed description about the trigger modes (https://www.matrix-vision/manuals/mvIMPACT Acquire API)

- C: TCameraTriggerMode
- C++: mvIMPACT::acquire::TCameraTriggerMode

**Note**

Trigger modes which use an external input (ctmOnLowLevel, ctmOnHighLevel, ctmOnRisingEdge, ctmOnFallingEdge) will use digital input 0 as input for the trigger signal. Input 0 is not restricted to the trigger function. It can always also be used as general purpose digital input. The input switching threshold of all inputs can be programmed with write_dac(level_in_mV). The best is to set this to the half of the input voltage. So for example if you apply a 24V switching signal to the digital inputs set the threshold to 12000 mV.

20.2.2.2 Exposure aka Integration

After an active trigger, the integration phase starts with a maximum jitter of \( t_{\text{trig}} \). If flash illumination is enabled in software the flash output will be activated exactly while the sensor chip is integrating light. Exposure time is adjustable by software in increments of \( t_{\text{readline}} \).

20.2.2.3 Readout

When integration is finished, the image is transferred to hidden storage cells on the CCD. Image data is then shifted out line-by-line and transferred to memory. Shifting out non active lines takes \( t_{\text{shift}} \), while shifting out active lines will consume \( t_{\text{readline}} \). The number of active pixels per line will not have any impact on readout speed.

20.2.3 CCD Timing
### Timings

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Pixel clock</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>20 MHz</td>
</tr>
<tr>
<td>t(_{\text{trig}})</td>
<td>Time from trigger (internal or external) to exposure start</td>
<td>3.6us</td>
</tr>
<tr>
<td>t(_{\text{trans}})</td>
<td>Image transfer time (move image to readout cells in CCD)</td>
<td>42.6us</td>
</tr>
<tr>
<td>t(_{\text{trans}})</td>
<td>time needed to readout a line</td>
<td>39.05us</td>
</tr>
<tr>
<td>t(_{\text{wait}})</td>
<td>time needed to shift unused lines away</td>
<td>3.6us</td>
</tr>
<tr>
<td>t(_{\text{exposure}})</td>
<td>Exposure time</td>
<td>1us..10s</td>
</tr>
<tr>
<td>t(_{\text{readline}})</td>
<td>time needed to readout a line</td>
<td>39.05us</td>
</tr>
<tr>
<td>t(_{\text{readline}})</td>
<td>time needed to shift unused lines away</td>
<td>3.6us</td>
</tr>
<tr>
<td>t(_{\text{wait}})</td>
<td>minimal time to next trigger</td>
<td>7.2us</td>
</tr>
</tbody>
</table>

\[ t_{\text{readout}} = (\text{ActiveLines} \times t_{\text{readline}}) + (504 - \text{ActiveLines}) \times t_{\text{vshift}} + t_{\text{readline}} \]

**Note**

In partial scan mode (readout window ysize < 480 lines).

To calculate the maximum frames per second ($FPS_{\text{max}}$) you will need following formula (Expose mode: No overlap):

\[
FPS_{\text{max}} = \frac{1}{t_{\text{trig}} + t_{\text{readout}} + t_{\text{exposure}} + t_{\text{trans}} + t_{\text{wait}}}
\]

(Expose mode: Overlapped):

\[
FPS_{\text{max}} = \frac{1}{t_{\text{trig}} + t_{\text{readout}} + t_{\text{trans}} + t_{\text{wait}} < t_{\text{exposure}}} \quad \text{or} \quad \frac{1}{t_{\text{trig}} + t_{\text{readout}} + t_{\text{trans}} + t_{\text{wait}} > t_{\text{exposure}}}
\]

20.2.3.1.1 Example: Frame rate as function of lines & exposure time

Now, when we insert the values using exposure time of, for example, 8000 us, 480 lines and 40MHz pixel clock (Expose mode: No overlap):

\[
FPS_{\text{max}} = \frac{1}{1.8 \text{ us} + ((480 \times 19.525 \text{ us}) + ((504 - 480) \times 1.80 \text{ us}) + 19.525 \text{ us}) + 8000 \text{ us} + 21.3 \text{ us} + 3.6 \text{ us}}
\]

\[
= 0.0000572690945899318068 \quad \text{1 / us}
\]

\[
= 57.3
\]

20.2.3.2 Frame rate calculator

**Note**

The calculator returns the max. frame rate supported by the sensor. Please keep in mind that it will depend on the interface and the used image format if this frame rate can be transferred.

**See also**

To find out how to achieve any defined freq. below or equal to the achievable max. freq., please have a look at Achieve a defined image frequency (HRTC) (p. [182](#)).
20.2.4 Reprogramming CCD Timing

Reprogramming the CCD Controller will happen when the following changes occur:

- Changing the exposure time
- Changing the capture window
- Changing Trigger Modes

Reprogram time consists of two phases:

1. Time needed to send data to the CCD controller depending on what is changed:
   - Exposure: about 2.3 ms
   - Window: about 4.6 ms
   - Trigger mode: from 5.90 ms, varies with oldmode/newmode combination
2. Time to initialize (erase) the CCD chip after reprogramming this is fixed, about 4.5 ms

So for example when reprogramming the capture window you will need (average values):

\[ t_{\text{regprog}} = \text{change\_window} + \text{init\_ccd} \]

\[ t_{\text{regprog}} = 5\text{ms} + 4.5\text{ms} \]

\[ t_{\text{regprog}} = 9.5\text{ms} \]

20.2.5 CCD Sensor Data

Device Structure:

- Interline CCD image sensor
- Image size: Diagonal 6mm (Type 1/3)
- Number of effective pixels: 659 (H) x 494 (V) approx. 330K pixels
- Total number of pixels: 692 (H) x 504 (V) approx. 350K pixels
- Chip size: 5.79mm (H) x 4.89mm (V)
- Unit cell size: 7.4um (H) x 7.4um (V)
- Optical black:
  - Horizontal (H) direction: Front 2 pixels, rear 31 pixels
  - Vertical (V) direction: Front 8 pixels, rear 2 pixels
- Number of dummy bits: Horizontal 16 Vertical 5
- Substrate material: Silicon
20.2.5.1 Characteristics

These zone definitions apply to both the color and gray scale version of the sensor.

![Zone Definition of Video Signal Shading](image)

20.2.5.2 Color version

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
<th>Measurement method</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>G Sensitivity</td>
<td>$S_g$</td>
<td>600</td>
<td>750</td>
<td></td>
<td>mV</td>
<td>1</td>
<td>1/30s accumulation</td>
</tr>
<tr>
<td>Sensitivity comparison</td>
<td>$R_r$</td>
<td>0.4</td>
<td>0.55</td>
<td>0.7</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$R_b$</td>
<td>0.3</td>
<td>0.45</td>
<td>0.6</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Saturation signal</td>
<td>$V_{sat}$</td>
<td>500</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>$T_a = 60^\circ C$</td>
</tr>
<tr>
<td>Smear</td>
<td>$S_m$</td>
<td></td>
<td>-100</td>
<td>-92</td>
<td>dB</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Video signal shading</td>
<td>$S_{Hg}$</td>
<td></td>
<td>20</td>
<td></td>
<td>%</td>
<td>4</td>
<td>Zone 0 and 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>25</td>
<td></td>
<td>%</td>
<td>4</td>
<td>Zone 0 to 11'</td>
</tr>
<tr>
<td>Uniformity between video signal channels</td>
<td>$\Delta S_{rg}$</td>
<td>8</td>
<td></td>
<td></td>
<td>%</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\Delta S_{bg}$</td>
<td>8</td>
<td></td>
<td></td>
<td>%</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Dark signal</td>
<td>$V_{dt}$</td>
<td>2</td>
<td></td>
<td></td>
<td>mV</td>
<td>6</td>
<td>$T_a = 60^\circ C$</td>
</tr>
<tr>
<td>Dark signal shading</td>
<td>$\Delta V_{dt}$</td>
<td>0.5</td>
<td></td>
<td></td>
<td>mV</td>
<td>7</td>
<td>$T_a = 60^\circ C$</td>
</tr>
<tr>
<td>Line crawl G</td>
<td>$L_{cg}$</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Line crawl R</td>
<td>$L_{cr}$</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Line crawl B</td>
<td>$L_{cb}$</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Lag</td>
<td>$L_{ag}$</td>
<td>0.5</td>
<td></td>
<td></td>
<td>%</td>
<td>9</td>
<td></td>
</tr>
</tbody>
</table>
20.2.5.3 Gray scale version

Image Sensor Characteristics

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
<th>Measurement method</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>S</td>
<td>700</td>
<td>880</td>
<td>mV</td>
<td>1</td>
<td>1/30s accumulation conversion value</td>
<td></td>
</tr>
<tr>
<td>Saturation signal</td>
<td>Vsat</td>
<td>500</td>
<td></td>
<td>mV</td>
<td>2</td>
<td>Ta = 60°C</td>
<td></td>
</tr>
<tr>
<td>Smear</td>
<td>Sm</td>
<td>-100</td>
<td>-92</td>
<td>dB</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Video signal shading</td>
<td>SH</td>
<td>20</td>
<td>%</td>
<td>4</td>
<td>Zone 0, 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dark signal</td>
<td>Vdt</td>
<td>2</td>
<td>mV</td>
<td>5</td>
<td>Ta = 60°C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dark signal shading</td>
<td>aVdt</td>
<td>0.5</td>
<td>mV</td>
<td>6</td>
<td>Ta = 60°C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lag</td>
<td>Lag</td>
<td>0.5</td>
<td>%</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Spectral Sensitivity Characteristics (Excludes lens characteristics and light source characteristics)
Measurement conditions

1) In the following measurements, the device drive conditions are at the typical values of the bias and clock voltage conditions.

2) In the following measurements, spot blemishes are excluded and, unless otherwise specified, the optical black level (OB) is used as the reference for the signal output, which is taken as the value measured at point [OB] of the measurement system.

Definition of standard imaging conditions

1) Standard imaging condition I:
   Use a pattern box (luminance: 700cd/m², color temperature of 3200K halogen source) as a subject. (pattern for evaluation is not applicable.) Use a testing standard lens with CM500S (t = 1.0mm) as an IR cut filter and image at F8. The luminous intensity to the sensor receiving surface at this point is defined as the standard sensitivity testing luminous intensity.

2) Standard imaging condition II:
   Image a light source (color temperature of 3200K) with a uniformity of brightness within 2% at all angles. Use a testing standard lens with CM500S (t = 1.0mm) as an IR cut filter. The luminous intensity is adjusted to the value indicated in each testing item by the lens diaphragm.

1. Sensitivity
   Set to standard imaging condition I. After selecting the electronic shutter mode with a shutter speed of 1/100s, measure the signal output (Vs) at the center of the screen, and substitute the value into the following formula.

   \[
   S = Vs \times \frac{250}{30} \text{ [mV]}
   \]

2. Saturation signal
   Set to standard imaging condition II. After adjusting the luminous intensity to 10 times the intensity with the average value of the signal output, 150mV, measure the minimum value of the signal output.

3. Smear
   Set to standard imaging condition II. With the lens diaphragm at F5.6 to F8, first adjust the luminous intensity to 500 times the intensity with the average value of signal output, 150mV. Then after the readout clock is stopped and the charge drain is executed by the electronic shutter at the respective H blankings, measure the maximum value (VSm [mV]) of the signal output and substitute the value into the following formula.

   \[
   Sm = 20 \times \log \left( \frac{VSm}{150} \times \frac{1}{500} \times \frac{1}{10} \right) \text{ [dB] (1/10V method conversion value)}
   \]

4. Video signal shading
   Set to standard imaging condition II. With the lens diaphragm at F5.6 to F8, adjust the luminous intensity so that the average value of the signal output is 150mV. Then measure the maximum (Vmax [mV]) and minimum (Vmin [mV]) values of the signal output and substitute the values into the following formula.

   \[
   SH = (Vmax - Vmin)/150 \times 100 \text{ [%]}
   \]

5. Dark signal
   Measure the average value of the signal output (Vdt [mV]) with the device ambient temperature 50°C and the device in the light-obstructed state, using the horizontal idle transfer level as a reference.
6. Dark signal shading
After measuring 5, measure the maximum (Vdmax [mV]) and minimum (Vdmin [mV]) values of the dark signal output and substitute the values into the following formula.

\[ \Delta V_{dt} = V_{dmax} - V_{dmin} [mV] \]

7. Lag
Adjust the signal output value generated by strobe light to 150mV. After setting the strobe light so that it strobes with the following timing, measure the residual signal (Vlag). Substitute the value into the following formula.

\[ \text{Lag} = \left( \frac{V_{lag}}{150} \right) \times 100 \% \]

Measurement System

Note) Adjust the amplifier gain so that the gain between [A] and [B] equals 1.

20.2.6 Device Feature And Property List

20.2.6.1 Gray scale version

20.2.6.2 Color version

MATRIX VISION GmbH
20.3 mvBlueFOX-[Model]221 (0.8 Mpix [1024 x 768])

20.3.1 Introduction

The CCD sensor is a highly programmable imaging module which will, for example, enable the following type of applications:

**Industrial applications:**

- triggered image acquisition with precise control of image exposure start by hardware trigger input.
- image acquisition of fast moving objects due to:
  - frame exposure, integrating all pixels at a time in contrast to CMOS imager which typically integrate line-by-line.
  - short shutter time, to get sharp images.
  - flash control output to have enough light for short time.

**Scientific applications:**

- long time exposure for low light conditions.
- optimizing image quality using the variable shutter control.

20.3.2 Details of operation

The process of getting an image from the CCD sensor can be separated into three different phases.

20.3.2.1 Trigger

When coming out of reset or ready with the last readout the CCD controller is waiting for a Trigger signal.

The following trigger modes are available:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous</td>
<td>Free running, no external trigger signal needed.</td>
</tr>
<tr>
<td>OnDemand</td>
<td>Image acquisition triggered by command (software trigger).</td>
</tr>
<tr>
<td>OnLowLevel</td>
<td>As long as trigger signal is Low camera acquires images with own timing.</td>
</tr>
<tr>
<td>OnHighLevel</td>
<td>As long as trigger signal is High camera acquires images with own timing.</td>
</tr>
<tr>
<td>OnFallingEdge</td>
<td>Each falling edge of trigger signal acquires one image.</td>
</tr>
<tr>
<td>OnRisingEdge</td>
<td>Each rising edge of trigger signal acquires one image.</td>
</tr>
<tr>
<td>OnHighExpos</td>
<td>Each rising edge of trigger signal acquires one image, exposure time corresponds to pulse width.</td>
</tr>
<tr>
<td>OnLowExpos</td>
<td>Each falling edge of trigger signal acquires one image, exposure time corresponds to pulse width.</td>
</tr>
<tr>
<td>OnAnyEdge</td>
<td>Start the exposure of a frame when the trigger input level changes from high to low or from low to high.</td>
</tr>
</tbody>
</table>
For detailed description about the trigger modes (https://www.matrix-vision/manuals/mvIMPACT Acquire API):

- **C**: TCameraTriggerMode
- **C++**: mvIMPACT::acquire::TCameraTriggerMode

### 20.3.2.2 Exposure aka Integration

After an active trigger, the exposure phase starts with a maximum jitter of $t_{\text{trig}}$. If flash illumination is enabled in software the flash output will be activated exactly while the sensor chip is integrating light. Integration time is adjustable by software in increments of $t_{\text{readline}}$.

### 20.3.2.3 Readout

When exposure is finished, the image is transferred to hidden storage cells on the CCD. Image data is then shifted out line-by-line and transferred to memory. Shifting out non active lines takes $tv_{\text{shift}}$, while shifting out active lines will consume $t_{\text{readline}}$. The number of active pixels per line will not have any impact on readout speed.

### 20.3.3 CCD Timing

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Pixel clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{\text{trig}}$</td>
<td>Time from trigger (internal or external) to exposure start</td>
<td>9.7us</td>
</tr>
<tr>
<td>$t_{\text{trans}}$</td>
<td>Image transfer time (move image to readout cells in CCD)</td>
<td>45us</td>
</tr>
<tr>
<td>$t_{\text{readline}}$</td>
<td>Time needed to readout a line</td>
<td>65.4us</td>
</tr>
<tr>
<td>$t_{\text{vshift}}$</td>
<td>Time needed to shift unused lines away</td>
<td>9.7us</td>
</tr>
<tr>
<td>$t_{\text{wait}}$</td>
<td>Minimal time to next trigger</td>
<td>116us</td>
</tr>
<tr>
<td>$t_{\text{exposure}}$</td>
<td>Integration time</td>
<td>1us..10s</td>
</tr>
<tr>
<td>$t_{\text{readout}}$</td>
<td>Image readout time (move image from readout cells to memory)</td>
<td>$t_{\text{readout}} = (\text{ActiveLines} \times t_{\text{readline}}) + (788 \times \text{ActiveLines}) \times t_{\text{vshift}} + t_{\text{readline}}$</td>
</tr>
</tbody>
</table>
In partial scan mode (readout window ysize < 768 lines).

To calculate the maximum frames per second (FPS$_{\text{max}}$) you will need following formula (Expose mode: Sequential):

$$\text{FPS}_{\text{max}} = \frac{1}{t_{\text{trig}} + t_{\text{readout}} + t_{\text{exposure}} + t_{\text{trans}} + t_{\text{wait}}}$$

(Expose mode: Overlapped):

$$\begin{align*}
\text{FPS}_{\text{max}} &= \frac{1}{t_{\text{trig}} + t_{\text{readout}} + t_{\text{trans}} + t_{\text{wait}} < t_{\text{exposure}}} \\
\text{FPS}_{\text{max}} &= \frac{1}{t_{\text{trig}} + t_{\text{readout}} + t_{\text{trans}} + t_{\text{wait}} > t_{\text{exposure}}}
\end{align*}$$

Example: Frame rate as function of lines & exposure time

Now, when we insert the values using exposure time of, for example, 8000 us, 768 lines and 40MHz pixel clock (Expose mode: Sequential):

$$\text{FPS}_{\text{max}} = \frac{1}{4.85 \text{ us} + ((768 \times 32.7 \text{ us}) + ((788 - 768) \times 4.85 \text{ us}) + 32.7 \text{ us}) + 8000 \text{ us} + 22.5 \text{ us} + 58 \text{ us}}$$

$$= \frac{1}{0.000030004215592290717 \text{ } \text{1 us}^{-1}}$$

$$= 30$$

The calculator returns the max. frame rate supported by the sensor. Please keep in mind that it will depend on the interface and the used image format if this frame rate can be transferred.

See also

To find out how to achieve any defined freq. below or equal to the achievable max. freq., please have a look at Achieve a defined image frequency (HRTC) (p. 182).

20.3.4 Reprogramming CCD Timing

Reprogramming the CCD Controller will happen when the following changes occur

- Changing the exposure time
- Changing the capture window
- Changing Trigger Modes

Reprogram time consists of two phases
1. Time needed to send data to the CCD controller depending on what is changed
   exposure: about 2.3 ms
   window: about 4.6 ms
   trigger mode: from 5.90 ms,
   varies with oldmode/newmode combination

2. Time to initialize (erase) the CCD chip after reprogramming this is fixed, about 4.5 ms

So for example when reprogramming the capture window you will need (average values)

\[ t_{\text{regprog}} = \text{change\_window} + \text{init\_ccd} \]
\[ t_{\text{regprog}} = 5\text{ms} + 4.5\text{ms} \]
\[ t_{\text{regprog}} = 9.5\text{ms} \]

20.3.5 CCD Sensor Data

Device Structure

- Interline CCD image sensor
- Image size: Diagonal 6 mm (Type 1/3)
- Number of effective pixels: 1025 (H) x 768 (V) approx. 790K pixels
- Total number of pixels: 1077 (H) x 788 (V) approx. 800K pixels
- Chip size: 5.80 mm (H) x 4.92 mm (V)
- Unit cell size: 4.65 um (H) x 4.65 um (V)
- Optical black:
  - Horizontal (H) direction: Front 3 pixels, rear 40 pixels
  - Vertical (V) direction: Front 7 pixels, rear 2 pixels
- Number of dummy bits: Horizontal 29 Vertical 1
- Substrate material: Silicon

20.3.5.1 Characteristics

These zone definitions apply to both the color and gray scale version of the sensor.

**Zone Definition of Video Signal Shading**

![Zone Definition Diagram](image-url)
20.3.5.2 Color version

**Image Sensor Characteristics**  
(Ta = 25°C)

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
<th>Measurement method</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>G sensitivity</td>
<td>Sg</td>
<td>320</td>
<td>400</td>
<td></td>
<td>mV</td>
<td>1</td>
<td>1/30s accumulation</td>
</tr>
<tr>
<td>Sensitivity comparison</td>
<td>R</td>
<td>0.4</td>
<td>0.55</td>
<td>0.7</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>0.3</td>
<td>0.45</td>
<td>0.6</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Saturation signal</td>
<td>Vsat</td>
<td>450</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>Ta = 60°C</td>
</tr>
<tr>
<td>Smear</td>
<td>Sm</td>
<td>0.001</td>
<td>0.004</td>
<td>%</td>
<td>3</td>
<td>No electronic shutter</td>
<td></td>
</tr>
<tr>
<td>Video signal shading</td>
<td>SHg</td>
<td></td>
<td></td>
<td>20</td>
<td>%</td>
<td>Zone 0 and 1</td>
<td></td>
</tr>
<tr>
<td>Uniformity between video</td>
<td>ΔSrG</td>
<td>8</td>
<td>%</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>signal channels</td>
<td>ΔSbg</td>
<td>8</td>
<td>%</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dark signal</td>
<td>Vdt</td>
<td>6</td>
<td>mV</td>
<td>6</td>
<td></td>
<td>Ta = 60°C, 20 frames/s</td>
<td></td>
</tr>
<tr>
<td>Dark signal shading</td>
<td>ΔVdt</td>
<td>2</td>
<td>mV</td>
<td>7</td>
<td></td>
<td>Ta = 60°C, 20 frames/s</td>
<td></td>
</tr>
<tr>
<td>Line crawl G</td>
<td>Lcg</td>
<td>3.8</td>
<td>%</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line crawl R</td>
<td>Lcr</td>
<td>3.8</td>
<td>%</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line crawl B</td>
<td>Lcb</td>
<td>3.8</td>
<td>%</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lag</td>
<td>Lag</td>
<td>0.5</td>
<td>%</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Spectral Sensitivity Characteristics**  
(excludes lens characteristics and light source characteristics)

![Spectral Sensitivity Characteristics Graph](image)
20.3.5.3 Gray scale version

### Image Sensor Characteristics

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
<th>Measurement Method</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>S</td>
<td>360</td>
<td>450</td>
<td></td>
<td>mV</td>
<td>1</td>
<td>1/30s accumulation</td>
</tr>
<tr>
<td>Saturation signal</td>
<td>Vsat</td>
<td>450</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>$T_a = 60^\circ$C</td>
</tr>
<tr>
<td>Smear</td>
<td>Sm</td>
<td>0.001</td>
<td>0.004</td>
<td>%</td>
<td></td>
<td>3</td>
<td>No electronic shutter</td>
</tr>
<tr>
<td>Video signal shading</td>
<td>SH</td>
<td></td>
<td></td>
<td>20</td>
<td>%</td>
<td>4</td>
<td>Zone 0 and I</td>
</tr>
<tr>
<td>Dark signal</td>
<td>Vdt</td>
<td>6</td>
<td></td>
<td>25</td>
<td>%</td>
<td>4</td>
<td>Zone 0 to Ii’</td>
</tr>
<tr>
<td>Dark signal shading</td>
<td>$\Delta$Vdt</td>
<td>2</td>
<td></td>
<td></td>
<td>mV</td>
<td>6</td>
<td>$T_a = 60^\circ$C, 20 frame/s</td>
</tr>
<tr>
<td>Lag</td>
<td>Lag</td>
<td>0.5</td>
<td></td>
<td></td>
<td>%</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

### Spectral Sensitivity Characteristics

(excludes lens characteristics and light source characteristics)
Measurement conditions

1) In the following measurements, the device drive conditions are at the typical values of the progressive scan mode, bias and clock voltage conditions.

2) In the following measurements, spot blemishes are excluded and, unless otherwise specified, the optical black level (0B) is used as the reference for the signal output, which is taken as the value measured at point [1B] of the measurement system.

Definition of standard imaging conditions

1) Standard imaging condition I:
   Use a pattern box (luminance: 706cd/m², color temperature of 3200K halogen source) as a subject. (Pattern for evaluation is not applicable.) Use a testing standard lens with CM500S (t = 1.0mm) as an IR cut filter and image at F8. The luminous intensity to the sensor receiving surface at this point is defined as the standard sensitivity testing luminous intensity.

2) Standard imaging condition II:
   Image a light source (color temperature of 3200K) with a uniformity of brightness within 2% at all angles. Use a testing standard lens with CM500S (t = 1.0mm) as an IR cut filter. The luminous intensity is adjusted to the value indicated in each testing item by the lens diaphragm.

3) Standard imaging condition III:
   Image a light source (color temperature of 3200K) with a uniformity of brightness within 2% at all angles. Use a testing standard lens (exit pupil distance -33mm) with CM500S (t = 1.0mm) as an IR cut filter. The luminous intensity is adjusted to the value indicated in each testing item by the lens diaphragm.

1. Sensitivity
   Set to standard imaging condition I. After selecting the electronic shutter mode with a shutter speed of 1/250s, measure the signal outputs (Vs) at the center of screen, and substitute the values into the following formulas.

   \[ S = V_s \times \frac{250}{30} \text{ [mV]} \]

2. Saturation signal
   Set to standard imaging condition II. After adjusting the luminous intensity to 10 times the intensity with the average value of the signal output, 150mV, measure the minimum values of the signal output.

3. Smear
   Set to standard imaging condition II. With the lens diaphragm at F5.6 to F8, first adjust the luminous intensity to 500 times the intensity with the average value of the signal output, 150mV. Then after the readout clock is stopped and the charge drain is executed by the electronic shutter at the respective H blankings, measure the maximum value (Vsm [mV]) of the signal output, and substitute the values into the following formula.

   \[ S_m = \frac{V_{sm}}{150} \times \frac{1}{500} \times \frac{1}{10} \times 100 \% \text{ (1/10V method conversion value)} \]
4. Video signal shading
   Set to standard imaging condition III. With the lens diaphragm at F5.6 to F8, adjust the luminous intensity
   so that the average value of the signal output is 150mV. Then measure the maximum (Vmax [mV]) and
   minimum (Vmin [mV]) values of the signal output and substitute the values into the following formula.

   \[ SH = \frac{(V_{\text{max}} - V_{\text{min}})}{150} \times 100 \% \]

5. Dark signal
   Measure the average value of the signal output (Vdt [mV]) with the device ambient temperature 60°C and
   the device in the light-obstructed state, using the horizontal idle transfer level as a reference.

6. Dark signal shading
   After measuring 5, measure the maximum (Vdmax [mV]) and minimum (Vdmin [mV]) values of the dark
   signal output and substitute the values into the following formula.

   \[ \Delta V_{dt} = V_{\text{dmax}} - V_{\text{dmin}} [\text{mV}] \]

7. Lag
   Adjust the signal output value generated by strobe light to 150mV. After setting the strobe light so that it
   strobos with the following timing, measure the residual signal (Vlag). Substitute the value into the following
   formula.

   \[ = \frac{(V_{\text{lag}}/150)}{100} \times 100 \% \]

---

**Measurement System**

CCD signal output [V]

\[ \xrightarrow{\text{C.O.S}} \]

\[ \xrightarrow{\text{AMP}} \]

\[ \xrightarrow{\text{S.H}} \]

Signal output [V]

---

**Note** Adjust the amplifier gain so that the gain between [V] and [V] equals 1.
20.3.6 CCD Signal Processing

The CCD signal is processed with an analog front-end and digitized by an 12 bit analog-to-digital converter (ADC). The analog front-end contains a programmable gain amplifier which is variable from 0db (gain=0) to 30dB (gain=255).

The 8 most significant bits of the ADC are captured to the frame buffer. This will give the following transfer function (based on the 8 bit digital code): Digital_code [lsb] = ccd_signal[V] \ast 256[lsb/V] \ast \exp(gain[bB]/20) lsb : least significant bit (smallest digital code change)

20.3.7 Device Feature And Property List

20.3.7.1 Gray scale version

20.3.7.2 Color version

20.4 mvBlueFOX-[Model]223 (1.4 Mpix [1360 x 1024])

20.4.1 Introduction

The CCD sensor is a highly programmable imaging module which will, for example, enable the following type of applications

**Industrial applications:**

- triggered image acquisition with precise control of image exposure start by hardware trigger input.
- image acquisition of fast moving objects due to:
  - frame exposure, integrating all pixels at a time in contrast to CMOS imager which typically integrate line-by-line.
  - short shutter time, to get sharp images.
  - flash control output to have enough light for short time.

**Scientific applications:**

- long time exposure for low light conditions.
- optimizing image quality using the variable shutter control.

20.4.2 Details of operation

The process of getting an image from the CCD sensor can be separated into three different phases.

20.4.2.1 Trigger

When coming out of reset or ready with the last readout the CCD controller is waiting for a Trigger signal.

The following trigger modes are available:
20.4.2 Exposure aka Integration

After an active trigger, the exposure phase starts with a maximum jitter of $t_{\text{trig}}$. If flash illumination is enabled in software the flash output will be activated exactly while the sensor chip is integrating light. Exposure time is adjustable by software in increments of $t_{\text{readline}}$.

20.4.2.3 Readout

When exposure is finished, the image is transferred to hidden storage cells on the CCD. Image data is then shifted out line-by-line and transferred to memory. Shifting out non active lines takes $t_{\text{shift}}$, while shifting out active lines will consume $t_{\text{readline}}$. The number of active pixels per line will not have any impact on readout speed.
20.4.3.1 Timings

**Note**

In partial scan mode (readout window ysize < 1024 lines).

To calculate the maximum frames per second (FPS\textsubscript{max}) you will need following formula (Expose mode: No overlap):

20.4.3.1.1 Example: Frame rate as function of lines & exposure time

Now, when we insert the values using exposure time of, for example, 8000 us, 1024 lines and 56MHz pixel clock (Expose mode: No overlap):

See also

To find out how to achieve any defined freq. below or equal to the achievable max. freq., please have a look at *Achieve a defined image frequency (HRTC)* (p. 182).

20.4.4 Reprogramming CCD Timing

Reprogramming the CCD Controller will happen when the following changes occur

- Changing the exposure time
- Changing the capture window
- Changing Trigger Modes

Reprogram time consists of two phases

1. Time needed to send data to the CCD controller depending on what is changed exposure: abt 2..3ms
   window: abt 4..6ms
   trigger mode: from 5..90ms, varies with oldmode/newmode combination

2. Time to initialize (erase) the CCD chip after reprogramming this is fixed, abt 4.5 ms

So for example when reprogramming the capture window you will need (average values)

\[ t_{\text{regprog}} = \text{change\_window} + \text{init\_ccd} \]

\[ t_{\text{regprog}} = 5\text{ms} + 4.5\text{ms} \]

\[ t_{\text{regprog}} = 9.5\text{ms} \]
20.4.5 CCD Sensor Data

Device Structure

- Interline CCD image sensor
- Image size: Diagonal 8mm (Type 1/2)
- Number of effective pixels: 1392 (H) x 1040 (V) approx. 1.45M pixels
- Total number of pixels: 1434 (H) x 1050 (V) approx. 1.5M pixels
- Chip size: 7.60mm (H) x 6.2mm (V)
- Unit cell size: 4.65um (H) x 4.65um (V)
- Optical black:
  - Horizontal (H) direction: Front 2 pixels, rear 40 pixels
  - Vertical (V) direction: Front 8 pixels, rear 2 pixels
- Number of dummy bits: Horizontal 20 Vertical 3
- Substrate material: Silicon

20.4.5.1 Characteristics

These zone definitions apply to both the color and gray scale version of the sensor.

Zone Definition of Video Signal Shading
### 20.4.5.2 Color version

**Image Sensor Characteristics** \(^{(Ta = 25\degree C)}\)

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
<th>Measurement method</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>G sensitivity</td>
<td>Sa</td>
<td>320</td>
<td>400</td>
<td></td>
<td>mV</td>
<td>1</td>
<td>1/30s accumulation</td>
</tr>
<tr>
<td>Sensitivity comparison</td>
<td>R</td>
<td>0.4</td>
<td>0.55</td>
<td>0.7</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>0.3</td>
<td>0.46</td>
<td>0.6</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Saturation signal</td>
<td>Vsat</td>
<td>450</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>(Ta = 60\degree C)</td>
</tr>
<tr>
<td></td>
<td>Vsat2</td>
<td>380</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>High frame rate readout mode</td>
</tr>
<tr>
<td></td>
<td>Vsat4</td>
<td>380</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>High frame rate readout two pixels addition^1</td>
</tr>
<tr>
<td>Smear</td>
<td>Sm</td>
<td>0.001</td>
<td>0.0025</td>
<td>%</td>
<td>3</td>
<td>Progressive scan readout, high frame rate readout two pixels addition^2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.002</td>
<td>0.005</td>
<td>%</td>
<td>3</td>
<td>High frame rate readout mode</td>
<td></td>
</tr>
<tr>
<td>Video signal shading</td>
<td>SHg</td>
<td>20</td>
<td></td>
<td></td>
<td>%</td>
<td>4</td>
<td>Zone 0 and 1</td>
</tr>
<tr>
<td>Uniformity between video signal channels</td>
<td>(\Delta Srg)</td>
<td>8</td>
<td>8</td>
<td>%</td>
<td>5</td>
<td>Zone 0 to 1'</td>
<td></td>
</tr>
<tr>
<td>Dark signal</td>
<td>Ydt</td>
<td>8</td>
<td></td>
<td></td>
<td>mV</td>
<td>6</td>
<td>(Ta = 60\degree C, 15) frames/s</td>
</tr>
<tr>
<td>Dark signal shading</td>
<td>(\Delta Ydt)</td>
<td>2</td>
<td></td>
<td></td>
<td>mV</td>
<td>7</td>
<td>(Ta = 60\degree C, 15) frames/s^2</td>
</tr>
<tr>
<td>Line crawl G</td>
<td>Leg</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Line crawl R</td>
<td>Lcr</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Line crawl B</td>
<td>Lcb</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Lag</td>
<td>Lag</td>
<td>0.5</td>
<td></td>
<td></td>
<td>%</td>
<td>9</td>
<td></td>
</tr>
</tbody>
</table>

^1 Vsat4 is the saturation signal amount at two pixels addition, and it is 190mV per one pixel. Vsus internal generation value ensures 190mV per one pixel of the saturation signal amount in high frame rate two pixels addition mode.

^2 Eliminates the dark signal shading in the vertical direction by the high-speed transfer of the vertical register.

---

**Spectral Sensitivity Characteristics** (excludes lens characteristics and light source characteristics)

![Spectral Sensitivity Characteristics](image-url)
20.4.5.3 Gray scale version

**Image Sensor Characteristics** (Ta = 25°C)

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
<th>Measurement method</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>S</td>
<td>350</td>
<td>450</td>
<td></td>
<td>mV</td>
<td>1/30s accumulation</td>
<td>Progressive scan readout mode</td>
</tr>
<tr>
<td>Saturation signal</td>
<td>Vsat</td>
<td>450</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>Ta = 60°C High frame rate readout mode</td>
</tr>
<tr>
<td></td>
<td>Vsat2</td>
<td>380</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>High frame rate readout</td>
</tr>
<tr>
<td></td>
<td>Vsat4</td>
<td>380</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>High frame rate readout two pixels addition</td>
</tr>
<tr>
<td>Smear</td>
<td>Sm</td>
<td>0.001</td>
<td>0.0025</td>
<td>%</td>
<td>3</td>
<td></td>
<td>Progressive scan readout, high frame rate readout two pixels addition</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.002</td>
<td>0.005</td>
<td>%</td>
<td>3</td>
<td></td>
<td>High frame rate readout mode</td>
</tr>
<tr>
<td>Video signal shading</td>
<td>SHg</td>
<td>20</td>
<td></td>
<td></td>
<td>%</td>
<td>4</td>
<td>Zone 0 and 1</td>
</tr>
<tr>
<td>Dark signal</td>
<td>Vdt</td>
<td>8</td>
<td></td>
<td></td>
<td>mV</td>
<td>5</td>
<td>Ta = 60°C, 15 frames/s</td>
</tr>
<tr>
<td>Dark signal shading</td>
<td>ΔVdt</td>
<td>2</td>
<td></td>
<td></td>
<td>mV</td>
<td>6</td>
<td>Ta = 60°C, 15 frames/s</td>
</tr>
<tr>
<td>Lag</td>
<td>Lag</td>
<td>0.5</td>
<td></td>
<td></td>
<td>%</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

*Vsat4 is the saturation signal amount at two pixels addition, and it is 190mV per one pixel. Vsat internal generation value ensures 190mV per one pixel of the saturation signal amount in high frame rate two pixels addition mode.*

*Eliminates the dark signal shading in the vertical direction by the high-speed transfer of the vertical register.*

**Spectral Sensitivity Characteristics** (excludes lens characteristics and light source characteristics)

![Spectral Sensitivity Graph](image_url)
1. G sensitivity, sensitivity comparison
   Set to standard imaging condition 1. After selecting the electronic shutter mode with a shutter speed of 1/100s, measure the signal outputs \( V_{Gr}, V_{Gb}, V_k \) and \( V_b \) at the center of each Gr, Gb, R and B channel screen, and substitute the values into the following formulas.

\[
\begin{align*}
V_0 &= \frac{(V_{Gr} + V_{Gb})}{2} \\
S_g &= \frac{V_G \times 100}{50} \, [\text{mV}] \\
R_t &= \frac{V_R}{V_0} \\
R_b &= \frac{V_b}{V_0}
\end{align*}
\]

2. Saturation signal
   Set to standard imaging condition 1. After adjusting the luminous intensity to 20 times the intensity with the average value of the Gr signal output, 150mV, measure the minimum values of the Gr, Gb, R and B signal outputs.

3. Smear
   Set to standard imaging condition 1. With the lens diaphragm at F5.6 to F8, first adjust the average value of the Gr signal output to 150mV. Measure the average values of the Gr signal output, Gb signal output, R signal output and B signal output \((G_{ra}, G_{ba}, R_a, B_a)\), and then adjust the luminous intensity to 500 times the intensity with the average value of the Gr signal output, 150mV. After the readout clock is stopped and the charge drain is executed by the electronic shutter at the respective H blankings, measure the maximum value \((V_{sm} \, [\text{mV}])\) independent of the Gr, Gb, R and B signal outputs, and substitute the values into the following formula.

\[
S_m = \frac{V_{sm} + G_{ra} + G_{ba} + R_a + B_a}{4} \times \frac{1}{\frac{500}{10}} \times 100 \, [%] \quad (1/10V \, \text{method conversion value})
\]
4. Video signal shading
Set to standard imaging condition I. With the lens diaphragm at F5.6 to F8, adjust the luminous intensity so that the average value of the Gr signal output is 150mV. Then measure the maximum (Grmax [mV]) and minimum (Grmin [mV]) values of the Gr signal output and substitute the values into the following formula.

\[ SHg = \frac{(Gr_{\text{max}} - Gr_{\text{min}})}{150} \times 100 \% \]

5. Uniformity between video signal channels
After measuring 4, measure the maximum (Rmax [mV]) and minimum (Rmin [mV]) values of the R signal and the maximum (Bmax [mV]) and minimum (Bmin [mV]) values of the B signal, and substitute the values into the following formulas.

\[ \Delta S_{\text{rgb}} = \frac{(R_{\text{max}} - R_{\text{min}})}{150} \times 100 \% \]
\[ \Delta S_{\text{rgb}} = \frac{(B_{\text{max}} - B_{\text{min}})}{150} \times 100 \% \]

6. Dark signal
Measure the average value of the signal output (Vdt [mV]) with the device ambient temperature 60°C and the device in the light-obstructed state, using the horizontal idle transfer level as a reference.

7. Dark signal shading
After measuring 6, measure the maximum (Vdmax [mV]) and minimum (Vdmin [mV]) values of the dark signal output and substitute the values into the following formula.

\[ \Delta V_{\text{d}} = V_{\text{dmax}} - V_{\text{dmin}} \] [mV]

8. Line crawl
Set to standard imaging condition I. Adjusting the luminous intensity so that the average value of the Gr signal output is 150mV, and then insert R, G and B filters and measure the difference between G signal lines (ΔGr, ΔGg, ΔGb [mV]) as well as the average value of the G signal output (Gsr, Gsg, Gsb). Substitute the values into the following formula.

\[ Lci = \frac{\Delta G_i}{G_{\text{sr}}} \times 100 \% (i = r, g, b) \]

9. Lag
Adjust the Gr signal output value generated by strobe light to 150mV. After setting the strobe light so that it strobes with the following timing, measure the residual signal (Vlag). Substitute the value into the following formula.

\[ \text{Lag} = \frac{(V_{\text{lag}}/150)}{100} \% \]

20.4.6 CCD Signal Processing

The CCD signal is processed with an analog front-end and digitized by an 12 bit analog-to-digital converter (A→DC). The analog front-end contains a programmable gain amplifier which is variable from 0db (gain=0) to 30dB (gain=255).
The 8 most significant bits of the ADC are captured to the frame buffer. This will give the following transfer function (based on the 8 bit digital code): Digital_code [lsb] = ccd_signal[V] * 256[lsb/V] * exp(gain[bB]/20) lsb : least significant bit (smallest digital code change)

20.4.7 Device Feature And Property List

20.4.7.1 Gray scale version

20.4.7.2 Color version

20.5 mvBlueFOX-[Model]224 (1.9 Mpix [1600 x 1200])

20.5.1 Introduction

The CCD sensor is a highly programmable imaging module which will, for example, enable the following type of applications

Industrial applications:

• triggered image acquisition with precise control of image exposure start by hardware trigger input.
• image acquisition of fast moving objects due to:
  – frame exposure, integrating all pixels at a time in contrast to CMOS imager which typically integrate line-by-line.
  – short shutter time, to get sharp images.
  – flash control output to have enough light for short time.

Scientific applications:

• long time exposure for low light conditions.
• optimizing image quality using the variable shutter control.

20.5.2 Details of operation

The process of getting an image from the CCD sensor can be separated into three different phases.

20.5.2.1 Trigger

When coming out of reset or ready with the last readout the CCD controller is waiting for a Trigger signal. The following trigger modes are available:
### 20.5.2.2 Timings

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Pixel clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{\text{trig}}$</td>
<td>Time from trigger (internal or external) to exposure start</td>
<td>10.2us</td>
</tr>
<tr>
<td>$t_{\text{trans}}$</td>
<td>Image transfer time (move image to readout cells in CCD)</td>
<td>96us</td>
</tr>
<tr>
<td>$t_{\text{readline}}$</td>
<td>time needed to readout a line</td>
<td>96us</td>
</tr>
<tr>
<td>$t_{\text{vshift}}$</td>
<td>time needed to shift unused lines away</td>
<td>10.2us</td>
</tr>
<tr>
<td>$t_{\text{wait}}$</td>
<td>minimal time to next trigger</td>
<td>316us</td>
</tr>
<tr>
<td>$t_{\text{exposure}}$</td>
<td>Exposure time</td>
<td>1us..10s</td>
</tr>
<tr>
<td>$t_{\text{readout}}$</td>
<td>Image readout time (move image from readout cells to memory)</td>
<td>$t_{\text{readout}} = (\text{ActiveLines} \times t_{\text{readline}}) + (1248 - \text{ActiveLines}) \times t_{\text{vshift}} + t_{\text{readline}}$</td>
</tr>
</tbody>
</table>

**Note**

In partial scan mode (readout window ysize < 1200 lines).

To calculate the maximum frames per second (FPS$_{\text{max}}$) you will need following formula (Expose mode: No overlap):

$$
\text{FPS}_{\text{max}} = \frac{1}{t_{\text{trig}} + t_{\text{readout}} + t_{\text{exposure}} + t_{\text{trans}} + t_{\text{wait}}}
$$
(Expose mode: Overlapped):

\[
\begin{align*}
\text{t}_{\text{trig}} + \text{t}_{\text{readout}} + \text{t}_{\text{trans}} + \text{t}_{\text{wait}} < \text{t}_{\text{exposure}}: & \quad \text{FPS}_{\text{max}} = \frac{1}{\text{t}_{\text{exposure}}} \\
\text{t}_{\text{trig}} + \text{t}_{\text{readout}} + \text{t}_{\text{trans}} + \text{t}_{\text{wait}} > \text{t}_{\text{exposure}}: & \quad \text{FPS}_{\text{max}} = \frac{1}{\text{t}_{\text{trig}} + \text{t}_{\text{readout}} + \text{t}_{\text{trans}} + \text{t}_{\text{wait}}}
\end{align*}
\]

20.5.2.2.1 Example: Frame rate as function of lines & exposure time

Now, when we insert the values using exposure time of, for example, 8000 us, 1200 lines and 40MHz pixel clock (Expose mode: No overlap):

\[
\begin{align*}
\text{FPS}_{\text{max}} &= \frac{1}{5.1 \text{ us} + ((1200 \times 48 \text{ us}) + ((1248 - 1200) \times 5.1 \text{ us}) + 48 \text{ us}) + 8000 \text{ us} + 48 \text{ us} + 158 \text{ us}} \\
&= 0.000015127700483632586 \quad 1 / \text{us} \\
&= 15.1
\end{align*}
\]

20.5.2.3 Frame rate calculator

**Note**

The calculator returns the max. frame rate supported by the sensor. Please keep in mind that it will depend on the interface and the used image format if this frame rate can be transferred.

**See also**

To find out how to achieve any defined freq. below or equal to the achievable max. freq., please have a look at *Achieve a defined image frequency (HRTC)* (p.182).

20.5.3 Reprogramming CCD Timing

Reprogramming the CCD Controller will happen when the following changes occur

- Changing the exposure time
- Changing the capture window
- Changing Trigger Modes

Reprogram time consists of two phases

1. Time needed to send data to the CCD controller depending on what is changed exposure : abt 2.3ms 
   window: abt 4.6ms trigger mode: from 5.90ms, varies with oldmode/newmode combination
2. Time to initialize (erase) the CCD chip after reprogramming this is fixed, abt 4.5 ms

So for example when reprogramming the capture window you will need (average values)

\[
\begin{align*}
\text{t}_{\text{regprog}} &= \text{change}_\text{window} + \text{init}_\text{ccd} \\
\text{t}_{\text{regprog}} &= 5\text{ms} + 4.5\text{ms} \\
\text{t}_{\text{regprog}} &= 9.5\text{ms}
\end{align*}
\]
20.5.4 CCD Sensor Data

Device Structure

- Interline CCD image sensor
- Image size: Diagonal 8.923mm (Type 1/1.8)
- Number of effective pixels: 1600 (H) x 1200 (V) approx. 1.92M pixels
- Total number of pixels: 1688 (H) x 1248 (V) approx. 2.11M pixels
- Chip size: 8.50mm (H) x 6.8mm (V)
- Unit cell size: 4.4um (H) x 4.4um (V)
- Optical black:
  - Horizontal (H) direction: Front 12 pixels, rear 48 pixels
  - Vertical (V) direction: Front 10 pixels, rear 2 pixels
- Number of dummy bits: Horizontal 28 Vertical 1
- Substrate material: Silicon

20.5.4.1 Characteristics

These zone definitions apply to both the color and gray scale version of the sensor.

[Diagram of Zone Definition of Video Signal Shading]
### Image Sensor Characteristics

* (Ta = 25°C)

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
<th>Measurement method</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>G Sensitivity</td>
<td>Sg</td>
<td>335</td>
<td>420</td>
<td>545</td>
<td>mV</td>
<td>1</td>
<td>1/30s accumulation</td>
</tr>
<tr>
<td>Sensitivity comparison</td>
<td>R</td>
<td>0.35</td>
<td>0.5</td>
<td>0.65</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>0.45</td>
<td>0.6</td>
<td>0.75</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Saturation signal</td>
<td>Vsat1</td>
<td>400</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>Ta = 60°C, No line addition&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>Vsat2&lt;sup&gt;2&lt;/sup&gt;</td>
<td>400</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2-line addition&lt;sup&gt;2&lt;/sup&gt;</td>
</tr>
<tr>
<td>Smear</td>
<td>Sm</td>
<td></td>
<td></td>
<td></td>
<td>dB</td>
<td>3</td>
<td>Progressive scan mode&lt;sup&gt;4&lt;/sup&gt;</td>
</tr>
<tr>
<td>Video signal shading</td>
<td>SH</td>
<td>20</td>
<td></td>
<td></td>
<td>%</td>
<td>4</td>
<td>Zone 0 and I</td>
</tr>
<tr>
<td></td>
<td></td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Zone 0 to II</td>
</tr>
<tr>
<td>Uniformity between video signal channels</td>
<td>ΔSr&lt;sub&gt;g&lt;/sub&gt;</td>
<td>8</td>
<td></td>
<td></td>
<td>%</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ΔSbg</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dark signal</td>
<td>Vdt</td>
<td>8</td>
<td></td>
<td></td>
<td>mV</td>
<td>6</td>
<td>Ta = 60°C, 14.955 frame/s</td>
</tr>
<tr>
<td>Dark signal shading</td>
<td>ΔVdt</td>
<td>2</td>
<td></td>
<td></td>
<td>mV</td>
<td>7</td>
<td>Ta = 60°C, 14.955 frame/s, &lt;sup&gt;9&lt;/sup&gt;</td>
</tr>
<tr>
<td>Line crawl G</td>
<td>Log</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Line crawl R</td>
<td>Lcr</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Line crawl B</td>
<td>Lcb</td>
<td>3.8</td>
<td></td>
<td></td>
<td>%</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Lag</td>
<td>Lag</td>
<td>0.5</td>
<td></td>
<td></td>
<td>%</td>
<td>9</td>
<td></td>
</tr>
</tbody>
</table>

<sup>1</sup> Vsat2 is the saturation signal level in 2-line addition mode, and is 200 mV per pixel.
<sup>2</sup> Progressive scan mode, 2/8-line readout mode, 2/4-line readout mode, and center scan modes (1) and (3).
<sup>3</sup> 2-line addition mode and center scan mode (2).
<sup>4</sup> Same for 2-line addition mode and center scan modes (2) and (3).
<sup>5</sup> Same for center scan mode (1).
<sup>6</sup> Same for AF modes (1) and (2).
<sup>9</sup> Excludes vertical dark signal shading caused by vertical register high-speed transfer.

### Spectral Sensitivity Characteristics

(Spectral sensitivity characteristics are excluded lens characteristics and light source characteristics)

[Graph showing spectral sensitivity characteristics]

MATRIX VISION GmbH
## Image Sensor Characteristics


ta = 25°C

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
<th>Measurement method</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>S</td>
<td>335</td>
<td>420</td>
<td>545</td>
<td>mV</td>
<td>1</td>
<td>1/30s accumulation</td>
</tr>
<tr>
<td>Saturation signal</td>
<td>Vsat</td>
<td>400</td>
<td></td>
<td></td>
<td>mV</td>
<td>2</td>
<td>Ta = 60°C</td>
</tr>
<tr>
<td></td>
<td>Vsat2*1</td>
<td>400</td>
<td></td>
<td></td>
<td>mV</td>
<td></td>
<td>No line addition*2</td>
</tr>
<tr>
<td>Smear</td>
<td>Sm</td>
<td></td>
<td></td>
<td>-100</td>
<td>-92</td>
<td>3</td>
<td>2-line addition*3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-94</td>
<td>-86</td>
<td></td>
<td>Progressive scan mode*4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-88</td>
<td>-80</td>
<td></td>
<td>2/4-line readout mode*5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2/8-line readout mode*6</td>
</tr>
<tr>
<td>Video signal shading</td>
<td>SH</td>
<td></td>
<td></td>
<td>20</td>
<td>%</td>
<td>4</td>
<td>Zone 0 and I</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>25</td>
<td></td>
<td></td>
<td>Zone 0 to II</td>
</tr>
<tr>
<td>Dark signal</td>
<td>Vdt</td>
<td></td>
<td></td>
<td>8</td>
<td>mV</td>
<td>5</td>
<td>Ta = 60°C, 14.985 frame/s</td>
</tr>
<tr>
<td>Dark signal shading</td>
<td>ΔVdt</td>
<td></td>
<td></td>
<td>2</td>
<td>mV</td>
<td>6</td>
<td>Ta = 60°C, 14.985 frame/s,</td>
</tr>
<tr>
<td>Lag</td>
<td>Lag</td>
<td></td>
<td></td>
<td>0.5</td>
<td>%</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

*1 Vsat2 is the saturation signal level in 2-line addition mode, and is 200 mV per pixel.
*2 Progressive scan mode, 2/8-line readout mode, 2/4-line readout mode, and center scan modes (1) and (3).
*3 2-line addition mode and center scan mode (2).
*4 Same for 2-line addition mode and center scan modes (2) and (3).
*5 Same for center scan mode (1).
*6 Same for AF modes (1) and (2).
*7 Excludes vertical dark signal shading caused by vertical register high-speed transfer.

## Spectral Sensitivity Characteristics (excludes lens characteristics and light source characteristics)

![Spectral Response Graph](image-url)
1. Sensitivity
   Set to the standard imaging condition I. After selecting the electronic shutter mode with a shutter speed of 1/100s, measure the signal output (Vs) at the center of the screen, and substitute the values into the following formulas.

   \[ S = V_s \times \frac{100}{30} \text{ [mV]} \]

2. Saturation signal
   Set to the standard imaging condition II. After adjusting the luminous intensity to 10 times the intensity with the average value of the G channel signal output, 150mV, measure the minimum values of the signal outputs.

3. Smear
   Set to standard imaging condition II. With the lens diaphragm at F5.6 to F8, adjust the luminous intensity to 500 times the intensity with the average value of the signal output, 150mV. After the readout clock is stopped and the charge drain is executed by the electronic shutter at the respective H blankings, measure the maximum value (Vsm [mV]) of the signal outputs, and substitute the values into the following formula. Smear in modes other than progressive scan mode is calculated from the storage time and signal addition method. As a result, 2-line addition mode and center scan modes (2) and (3) are the same as progressive scan mode, 2/4-line readout mode and center scan mode (1) are two times progressive scan mode, and 2/6-line readout mode and AF modes (1) and (2) are four times progressive scan mode.

   \[ Sm = 20 \times \log \left( \frac{V_{sm}}{200} \times \frac{1}{500} \times \frac{1}{10} \right) \text{ [dB]} \text{ (1/10 V method conversion value)} \]

4. Video signal shading
   Set to the standard imaging condition II. With the lens diaphragm at F5.6 to F8, adjusting the luminous intensity so that the average value of the signal output is 150mV. Then measure the maximum value (Vmax [mV]) and minimum value (Vmin [mV]) of the G signal output and substitute the values into the following formula.

   \[ SH = (V_{max} - V_{min})/150 \times 100 \text{ [%]} \]

5. Dark signal
   Measure the average value of the signal output (Vdt [mV]) with the device ambient temperature of 60°C and the device in the light-obstructed state, using the horizontal idle transfer level as a reference.

6. Dark signal shading
   After measuring 5, measure the maximum (Vdmax [mV]) and minimum (Vdmin [mV]) values of the dark signal output and substitute the values into the following formula.

   \[ \Delta Vdt = V_{dmax} - V_{dmin} \text{ [mV]} \]

7. Lag
   Adjust the signal output generated by the strobe light to 150mV. After setting the strobe light so that it strobos with the following timing, measure the residual signal amount (Vlag). Substitute the value into the following formula.
20.5. 20.5 mvBlueFOX-[Model]224 (1.9 Mpix [1600 x 1200])

\[ L_{ag} = \left( \frac{V_{ag}}{150} \right) \times 100\% \]

**20.5.5 CCD Signal Processing**

The CCD signal is processed with an analog front-end and digitized by an 12 bit analog-to-digital converter (A\rightarrow DC). The analog front-end contains a programmable gain amplifier which is variable from 0db (gain=0) to 30dB (gain=255).

\[ \text{Digital code [lsb]} = \text{ccd signal[V]} \times 256\text{[lsb/V]} \times \exp(\text{gain[bB]}/20) \]

\[ \text{lsb : least significant bit (smallest digital code change)} \]

**20.5.6 Device Feature And Property List**

20.5.6.1 Gray scale version

20.5.6.2 Color version
21 Appendix A.5 CMOS specific camera / sensor data

21.1 mvBlueFOX-[Model]200w (0.4 Mpix [752 x 480])

21.1.1 Introduction

The CMOS sensor module (MT9V034) incorporates the following features:

- resolution to 752 x 480 gray scale or RGB Bayer mosaic
- supports window AOI mode with faster readout
- high dynamic range \( \text{p.}[163] \) 110 dB
- programmable analog gain (0..12 dB)
- progressive scan sensor (no interlaced problems!)
- full frame shutter
- programmable readout timing with free capture windows and partial scan
- many trigger modes (free-running, hardware-triggered)

21.1.2 Details of operation

The sensor uses a full frame shutter (ShutterMode = "FrameShutter"), i.e. all pixels are reset at the same time and the exposure commences. It ends with the charge transfer of the voltage sampling. Furthermore, the sensor offers two different modes of operation:

- free running mode (Overlapping exposure and readout)
- snapshot mode (Sequential exposure and readout)

21.1.2.1 Free running mode

In free running mode, the sensor reaches its maximum frame rate. This is done by overlapping erase, exposure and readout phase. The sensor timing in free running mode is fixed, so there is no control when to start an acquisition. This mode is used with trigger mode Continuous.

To calculate the maximum frames per second (FPS\(_{\text{max}}\)) in free running mode you will need following formula:

\[
\text{FrameTime} = (\text{ImageWidth} + 61) \times ((\text{ImageHeight} + 45) / \text{PixelClock})
\]

If exposure time is lower than frame time:

\[
\text{FPS}_{\text{max}} = \frac{1}{\text{FrameTime}}
\]

If exposure time is greater than frame time:

\[
\text{FPS}_{\text{max}} = \frac{1}{\text{ExposureTime}}
\]
21.1.2.2 Snapshot mode

In snapshot mode, the image acquisition process consists of several sequential phases:

21.1.2.2.1 Trigger

Snapshot mode starts with a trigger. This can be either a hardware or a software signal.

The following trigger modes are available:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous</td>
<td>Free running, no external trigger signal needed.</td>
</tr>
<tr>
<td>OnDemand</td>
<td>Image acquisition triggered by command (software trigger).</td>
</tr>
<tr>
<td>OnLowLevel</td>
<td>As long as trigger signal is Low camera acquires images with own timing.</td>
</tr>
<tr>
<td>OnHighLevel</td>
<td>As long as trigger signal is High camera acquires images with own timing.</td>
</tr>
</tbody>
</table>

See also

Using external trigger with CMOS sensors (p. 162)

21.1.2.2.2 Erase, exposure and readout

All pixels are light sensitive at the same period of time. The whole pixel core is reset simultaneously and after the exposure time all pixel values are sampled together on the storage node inside each pixel. The pixel core is read out line-by-line after exposure.

Note

Exposure and read out cycle is carry-out in serial; that causes that no exposure is possible during read out.

The step width for the exposure time is 1 us.

Image data is then shifted out line-by-line and transferred to memory.

To calculate the maximum frames per second (FPS_{max}) in snapshot mode you will need following formula:

\[
\text{FrameTime} = (\text{ImageWidth} + 61) \times (\text{ImageHeight} + 45) / \text{PixelClock}
\]

\[
\text{FPS}_{\text{max}} = \frac{1}{\text{FrameTime} + \text{ExposureTime}}
\]

21.1.3 Measured frame rates

<table>
<thead>
<tr>
<th>AOI</th>
<th>PixelClock (MHz)</th>
<th>Exposure Time (us)</th>
<th>Maximal Frame Rate (fps)</th>
<th>PixelFormat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum</td>
<td>40</td>
<td>100</td>
<td>93.7</td>
<td>Mono8</td>
</tr>
<tr>
<td>W:608 x H:388</td>
<td>40</td>
<td>100</td>
<td>131.4</td>
<td>Mono8</td>
</tr>
<tr>
<td>W:492 x H:314</td>
<td>40</td>
<td>100</td>
<td>158.5</td>
<td>Mono8</td>
</tr>
<tr>
<td>W:398 x H:206</td>
<td>40</td>
<td>100</td>
<td>226.7</td>
<td>Mono8</td>
</tr>
</tbody>
</table>

MATRIX VISION GmbH
21.1.4 Sensor Data

Device Structure

- Progressive scan CMOS image sensor
- Image size: 4.51(H)x2.88(V)mm (Type 1/3")
- Number of effective pixels: 752 (H) x 480 (V)
- Unit cell size: 6um (H) x 6um (V)

21.1.4.1 Characteristics

21.1.4.2 Color version

![Color version graph]

21.1.4.3 Gray scale version

![Gray scale version graph]
21.2.1 Introduction

The CMOS sensor module (MT9M001) incorporates the following features:

- resolution to 1280 x 1024 gray scale
- supports window AOI mode with faster readout
- dynamic range 61dB
- programmable analog gain (0..12dB)
- progressive scan sensor (no interlaced problems!)
- rolling shutter
- programmable readout timing with free capture windows and partial scan
- many trigger modes (free-running, hardware-triggered)

21.2.2 Details of operation

The sensor uses following acquisition mode:

- **rolling shutter** (ShutterMode = "ElectronicRollingShutter").

With the **rolling shutter** the lines are exposed for the same duration, but at a slightly different point in time.

**Note**

Moving objects together with a rolling shutter can cause a shear in moving objects.

Furthermore, the sensor offers one operating mode:

- snapshot mode (which means sequential exposure and readout)

21.2.2.1 Snapshot mode

In snapshot mode, the image acquisition process consists of several sequential phases:

21.2.2.1.1 Trigger

Snapshot mode starts with a trigger. This can be either a hardware or a software signal.

The following trigger modes are available:
<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous</td>
<td>Free running, no external trigger signal needed.</td>
</tr>
<tr>
<td>OnDemand</td>
<td>Image acquisition triggered by command (software trigger).</td>
</tr>
<tr>
<td>OnLowLevel</td>
<td>As long as trigger signal is Low camera acquires images with own timing.</td>
</tr>
<tr>
<td>OnHighLevel</td>
<td>As long as trigger signal is High camera acquires images with own timing.</td>
</tr>
<tr>
<td>OnFallingEdge</td>
<td>Each falling edge of trigger signal acquires one image.</td>
</tr>
<tr>
<td>OnRisingEdge</td>
<td>Each rising edge of trigger signal acquires one image.</td>
</tr>
<tr>
<td>OnHighExposure</td>
<td>Each rising edge of trigger signal acquires one image, exposure time</td>
</tr>
<tr>
<td></td>
<td>corresponds to pulse width.</td>
</tr>
<tr>
<td>OnLowExposure</td>
<td>Each falling edge of trigger signal acquires one image, exposure time</td>
</tr>
<tr>
<td></td>
<td>corresponds to pulse width.</td>
</tr>
<tr>
<td>OnAnyEdge</td>
<td>Start the exposure of a frame when the trigger input level changes from</td>
</tr>
<tr>
<td></td>
<td>high to low or from low to high.</td>
</tr>
</tbody>
</table>

See also

For detailed description about the trigger modes (https://www.matrix-vision/manuals/[mvIMPACT Acquire API])

- C: TCameraTriggerMode
- C++: mvIMPACT::acquire::TCameraTriggerMode

21.2.2.1.2 Erase, exposure and readout

After the trigger pulse, the complete sensor array is erased. This takes some time, so there is a fix delay from about 285 us between the trigger pulse on digital input 0 and the start of exposure of the first line.

The exact time of exposure start of each line (except the first line) depends on the exposure time and the position of the line. The exposure of a particular line N is finished when line N is ready for readout. Image data is read out line-by-line and transferred to memory (see: http://www.matrix-vision.com/tl_files/mv11/←Glossary/art_rolling_shutter_en.pdf).

Exposure time is adjustable by software and depends on the image width. To calculate the exposure step size you will need following formula:

\[
\text{LineDelay} = 0
\]

\[
\text{PixelClkPeriod} = \frac{1}{\text{PixelClk}}
\]

\[
\text{RowTime} = (\text{ImageWidth} + 244 + \text{LineDelay}) \times \text{PixelClkPeriod}
\]

\[
\text{RowTime} = \text{MinExposurTime} = \text{ExposureStepSize}
\]

Image data is then shifted out line-by-line and transferred to memory.

To calculate the maximum frames per second (FPS\textsubscript{max}) in snapshot mode you will need following formula:

\[
\text{FrameTime} = (\text{ImageWidth} + 244) \times ((\text{ImageHeight} + 16) / \text{PixelClock})
\]

\[
\text{FPS}_{\text{max}} = \frac{1}{\text{FrameTime} + \text{ExposureTime}}
\]
21.2.2.2 CMOS Timing in Snapshot mode

![CMOS Timing Diagram]

21.2.3 Sensor Data

**Device Structure**

- Progressive scan CMOS image sensor
- Image size: 6.66(H)x5.32(V)mm (Type 1/2”)
- Number of effective pixels: 1280 (H) x 1024 (V)
- Unit cell size: 5.2um (H) x 5.2um (V)

21.2.4 Characteristics

21.2.4.1 Gray scale version

![Quantum Efficiency - Monochrome Graph]

**Quantum Efficiency - Monochrome**

- **FILTER IR-CUT**
- **FILTER DL-CUT**
21.2.5 Device Feature And Property List

21.2.5.1 Gray scale version

21.3 mvBlueFOX-[Model]202b (1.2 Mpix [1280 x 960])

21.3.1 Introduction

The CMOS sensor module (MT9M021) incorporates the following features:

- resolution to 1280 x 960 gray scale or RGB Bayer mosaic
- supports window AOI mode with faster readout
- programmable analog gain (0..12 dB)
- progressive scan sensor (no interlaced problems!)
- pipelined global shutter
- programmable readout timing with free capture windows and partial scan
- many trigger modes (free-running, hardware-triggered)

21.3.2 Details of operation

The sensor uses a pipelined global snapshot shutter (ShutterMode = "FrameShutter"), i.e. light exposure takes place on all pixels in parallel, although subsequent readout is sequential. Therefore the sensor offers two different modes of operation:

- free running mode (Overlapping exposure and readout)
- snapshot mode (Sequential exposure and readout)

21.3.2.1 Free running mode

In free running mode, the sensor reaches its maximum frame rate. This is done by overlapping erase, exposure and readout phase. The sensor timing in free running mode is fixed, so there is no control when to start an acquisition. This mode is used with trigger mode Continuous.

To calculate the maximum frames per second (FPS\text{max}) in free running mode you will need following formula:

\[
\text{FrameTime} = (\text{ImageHeight} \times (1650 / \text{PixelClock})) + (25 \times (1650 / \text{PixelClock}))
\]

If exposure time is lower than frame time:
21.3.2.2 Snapshot mode

In snapshot mode, the image acquisition process consists of several sequential phases:

21.3.2.2.1 Trigger

Snapshot mode starts with a trigger. This can be either a hardware or a software signal.

The following trigger modes are available:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous</td>
<td>Free running, no external trigger signal needed.</td>
</tr>
<tr>
<td>OnLowLevel</td>
<td>As long as trigger signal is <em>Low</em> camera acquires images with own timing.</td>
</tr>
<tr>
<td>OnHighLevel</td>
<td>As long as trigger signal is <em>High</em> camera acquires images with own timing.</td>
</tr>
</tbody>
</table>

See also

*Using external trigger with CMOS sensors* (p. 162)
21.3.2.2 Erase, exposure and readout

All pixels are light sensitive at the same period of time. The whole pixel core is reset simultaneously and after the exposure time all pixel values are sampled together on the storage node inside each pixel. The pixel core is read out line-by-line after exposure.

**Note**

Exposure and read out cycle is carry-out in serial; that causes that no exposure is possible during read out.

The step width for the exposure time is 1 us.

Image data is then shifted out line-by-line and transferred to memory.

To calculate the maximum frames per second (FPS\(_{\text{max}}\)) in snapshot mode you will need following formula:

\[
\text{FrameTime} = (\text{ImageHeight} \times \frac{1650}{\text{PixelClock}}) + (25 \times \frac{1650}{\text{PixelClock}})
\]

\[
\text{FPS}_{\text{max}} = \frac{1}{\text{FrameTime} + \text{ExposureTime}}
\]

### 21.3.3 Measured frame rates

<table>
<thead>
<tr>
<th>AOI</th>
<th>PixelClock (MHz)</th>
<th>Exposure Time (us)</th>
<th>Maximal Frame Rate (fps)</th>
<th>PixelFormat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum</td>
<td>40</td>
<td>100</td>
<td>24.6</td>
<td>Mono8</td>
</tr>
<tr>
<td>W:1036 x H:776</td>
<td>40</td>
<td>100</td>
<td>30.3</td>
<td>Mono8</td>
</tr>
<tr>
<td>W:838 x H:627</td>
<td>40</td>
<td>100</td>
<td>37.1</td>
<td>Mono8</td>
</tr>
<tr>
<td>W:678 x H:598</td>
<td>40</td>
<td>100</td>
<td>38.9</td>
<td>Mono8</td>
</tr>
<tr>
<td>W:550 x H:484</td>
<td>40</td>
<td>100</td>
<td>47.6</td>
<td>Mono8</td>
</tr>
</tbody>
</table>

21.3.4 Sensor Data

**Device Structure**

- CMOS image sensor (Type 1/3”)
- Number of effective pixels: 1280 (H) x 960 (V)
- Unit cell size: 3.75um (H) x 3.75um (V)
21.3.4.2 Color version

21.3.4.3 Gray scale version

21.3.5 Device Feature And Property List

21.3.5.1 Gray scale version

21.3.5.2 Color version
21.4  mvBlueFOX-[Model]202d (1.2 Mpix [1280 x 960])

21.4.1  Introduction

The CMOS sensor module (MT9M024) incorporates the following features:

- resolution to 1280 x 960 gray scale or RGB Bayer mosaic
- supports window AOI mode with faster readout
- programmable analog gain (0..12 dB)
- progressive scan sensor (no interlaced problems!)
- high dynamic range (p.167) 115 dB (with gray scale version)
- rolling shutter
- programmable readout timing with free capture windows and partial scan
- many trigger modes (free-running, hardware-triggered)

21.4.2  Details of operation

The sensor uses following acquisition mode:

- rolling shutter (ShutterMode = "ElectronicRollingShutter") With the rolling shutter the lines are exposed for the same duration, but at a slightly different point in time.

**Note**

Moving objects together with a rolling shutter can cause a shear in moving objects.

Furthermore, the sensor offers following operating modes:

- free running mode (Overlapping exposure and readout)
- snapshot mode (Sequential exposure and readout)

21.4.2.1  Free running mode

In free running mode, the sensor reaches its maximum frame rate. This is done by overlapping erase, exposure and readout phase. The sensor timing in free running mode is fixed, so there is no control when to start an acquisition. This mode is used with trigger mode **Continuous**.

To calculate the maximum frames per second (FPS\(_{\text{max}}\)) in free running mode you will need following formula:

\[
\text{FrameTime} = (\text{ImageHeight} \times (1650 / \text{PixelClock})) + (25 \times (1650 / \text{PixelClock})
\]

If exposure time is lower than frame time:

\[
\text{FPS\(_{\text{max}}\)} = \frac{1}{\text{FrameTime}}
\]

If exposure time is greater than frame time:

\[
\text{FPS\(_{\text{max}}\)} = \frac{1}{\text{ExposureTime}}
\]
21.4.2.2 Snapshot mode

In snapshot mode, the image acquisition process consists of several sequential phases:

21.4.2.2.1 Trigger

Snapshot mode starts with a trigger. This can be either a hardware or a software signal.

The following trigger modes are available:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous</td>
<td>Free running, no external trigger signal needed.</td>
</tr>
<tr>
<td>OnLowLevel</td>
<td>As long as trigger signal is Low camera acquires images with own timing.</td>
</tr>
<tr>
<td>OnHighLevel</td>
<td>As long as trigger signal is High camera acquires images with own timing.</td>
</tr>
</tbody>
</table>

See also

Using external trigger with CMOS sensors (p. 162)

21.4.2.2.2 Erase, exposure and readout

All pixels are light sensitive at the same period of time. The whole pixel core is reset simultaneously and after the exposure time all pixel values are sampled together on the storage node inside each pixel. The pixel core is read out line-by-line after exposure.

**Note**

Exposure and read out cycle is carry-out in serial; that causes that no exposure is possible during read out.

The step width for the exposure time is 1 us.

Image data is then shifted out line-by-line and transferred to memory.

To calculate the maximum frames per second (FPS\(_{\text{max}}\)) in snapshot mode you will need following formula:

\[
\text{FrameTime} = (\text{ImageHeight} \times (1650 / \text{PixelClock})) + (25 \times (1650 / \text{PixelClock}))
\]

\[
\text{FPS}_{\text{max}} = \frac{1}{\text{FrameTime} + \text{ExposureTime}}
\]

21.4.3 Measured frame rates

<table>
<thead>
<tr>
<th>AOI</th>
<th>PixelClock (MHz)</th>
<th>Exposure Time (us)</th>
<th>Maximal Frame Rate (fps)</th>
<th>PixelFormat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum</td>
<td>40</td>
<td>100</td>
<td>24.6</td>
<td>Mono8</td>
</tr>
<tr>
<td>W:1036 x H:776</td>
<td>40</td>
<td>100</td>
<td>30.3</td>
<td>Mono8</td>
</tr>
<tr>
<td>W:838 x H:627</td>
<td>40</td>
<td>100</td>
<td>37.1</td>
<td>Mono8</td>
</tr>
<tr>
<td>W:678 x H:598</td>
<td>40</td>
<td>100</td>
<td>38.9</td>
<td>Mono8</td>
</tr>
<tr>
<td>W:550 x H:484</td>
<td>40</td>
<td>100</td>
<td>47.6</td>
<td>Mono8</td>
</tr>
</tbody>
</table>
21.4.4 Sensor Data

Device Structure

- CMOS image sensor (Type 1/3")
- Number of effective pixels: 1280 (H) x 960 (V)
- Unit cell size: 3.75um (H) x 3.75um (V)

21.4.4.1 Characteristics

21.4.4.2 Color version

![Graph showing quantum efficiency vs. wavelength for red, green, blue, and IR-CUT filters.]

21.4.4.3 Gray scale version

![Graph showing quantum efficiency vs. wavelength for IR-CUT and DL-CUT filters.]
21.5.1 Introduction

The CMOS sensor module (MT9P031) incorporates the following features:

- resolution to 2592 x 1944 gray scale or RGB Bayer mosaic
- supports window AOI mode with faster readout
- programmable analog gain (0..32dB)
- progressive scan sensor (no interlaced problems!)
- rolling shutter / global reset release
- programmable readout timing with free capture windows and partial scan
- many trigger modes (free-running, hardware-triggered)

21.5.2 Details of operation

The sensor uses two acquisition modes:

- **rolling shutter** (ShutterMode = "ElectronicRollingShutter") and
- **global reset release shutter** (ShutterMode = "GlobalResetRelease").

With the **rolling shutter** the lines are exposed for the same duration, but at a slightly different point in time:
Note

Moving objects together with a rolling shutter can cause a shear in moving objects.

The global reset release shutter, which is only available in triggered operation, starts the exposure of all rows simultaneously and the reset to each row is released simultaneously, too. However, the readout of the lines is equal to the readout of the rolling shutter: line by line:

Note

This means, the bottom lines of the sensor will be exposed to light longer! For this reason, this mode will only make sense, if there is no extraneous light and the flash duration is shorter or equal to the exposure time.

Furthermore, the sensor offers two operating modes:

- free running mode (Overlapping exposure and readout)
- snapshot mode (Sequential exposure and readout) in triggered operation

21.5.2.1 Free running mode

In free running mode, the sensor reaches its maximum frame rate. This is done by overlapping erase, exposure and readout phase. The sensor timing in free running mode is fixed, so there is no control when to start an acquisition. This mode is used with trigger mode Continuous.

To calculate the maximum frames per second (FPS$_{\text{max}}$) in free running mode you will need following formula:

$$\text{FrameTime} = (\text{ImageWidth} + 900) \times ((\text{ImageHeight} + 9) / \text{PixelClock})$$

If exposure time is lower than frame time:

$$\text{FPS}_{\text{max}} = \frac{1}{\text{FrameTime}}$$

If exposure time is greater than frame time:

$$\text{FPS}_{\text{max}} = \frac{1}{\text{ExposureTime}}$$

21.5.2.2 Snapshot mode

In snapshot mode, the image acquisition process consists of several sequential phases:

21.5.2.2.1 Trigger

Snapshot mode starts with a trigger. This can be either a hardware or a software signal.

The following trigger modes are available:
21.5.2.2.2 Erase, exposure and readout

All pixels are light sensitive at the same period of time. The whole pixel core is reset simultaneously and after the exposure time all pixel values are sampled together on the storage node inside each pixel. The pixel core is read out line-by-line after exposure.

**Note**

Exposure and read out cycle is carry-out in serial; that causes that no exposure is possible during read out.

The step width for the exposure time is 1 us.

Image data is then shifted out line-by-line and transferred to memory.

To calculate the maximum frames per second (FPS\textsubscript{max}) in snapshot mode you will need following formula:

\[
\text{FrameTime} = (\text{ImageWidth} + 900) \times ((\text{ImageHeight} + 9) / \text{PixelClock})
\]

\[
\text{FPS}_{\text{max}} = \frac{1}{\text{FrameTime} + \text{ExposureTime}}
\]

21.5.2.3 Use cases

As mentioned before, "Global reset release" will only make sense, if a flash is used which is brighter than the ambient light. The settings in \textit{wxPropView} (p. 78) will look like this:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlashMode</td>
<td>&quot;DigOut0&quot;</td>
</tr>
<tr>
<td>FlashType</td>
<td>RollingShutterFlash</td>
</tr>
<tr>
<td>TestMode</td>
<td>Off</td>
</tr>
<tr>
<td>ShutterMode</td>
<td>GlobalResetRelease</td>
</tr>
</tbody>
</table>

In this case, \textit{DigOut0} gets a high signal as long as the exposure time (which is synchronized with the GlobalResetRelease). This signal can start a flash light.

21.5.3 Measured frame rates
### Sensor Data

#### Device Structure

- Progressive scan CMOS image sensor
- Image size: 5.70(H)x4.28(V)mm (Type 1/2.5")
- Number of effective pixels: 2592 (H) x 1944 (V)
- Unit cell size: 2.2um (H) x 2.2um (V)

#### Characteristics

#### Color version

![Graph showing quantum efficiency across wavelengths with a filter labeled IR-CUT]
21.5.4.3 Gray scale version

21.5.5 Device Feature And Property List

21.5.5.1 Gray scale version

21.5.5.2 Color version