XIS Reference Book
X-Ray Imaging Software

Table of Contents

List of Figures

List of Tables

1   General

1.1   References

Document Name	Document #
XRpad 4336 Reference Manual	27505
XRpad 4343 Reference Manual	39256

Table 1   References

1.2   About the program XIS

The program XIS **X**-Ray **I**maging **S**oftware is capable to demonstrate the functions of the digital amorphous Silicon **X-R**ay **D**etector (former **RID R**adiation **I**mage **D**etector) and XRpad series detectors. It can be used with any type of XRD/XRpad in combination with the installed PCI, PCIe or PCI-X frame grabber boards and with X-Ray detectors with Gigabit Ethernet /WLAN interface. It automatically detects the size of the sensor and receives RAW 16bit Grayscale images from the detector with up to 65535 Gray levels. The images are displayed on the screen with 256 grey levels.

Offset, **Gain/Offset** and **Gain Sequence/ Offset Images** can be used to correct the image data for the specific dark current of the detector and the irradiation of a specific x-ray source. The use of the Offset, Gain/Offset and Gain Sequence/ Offset Images is recommended for best image quality. A **Pixel Correction** allows the 'software repair' of defect pixels to enhance the image quality.

The detector allows the setting of different frame times. The shortest possible frame time in Free-Running-mode is selected at startup, but any timing can be set to optimize the system capabilities.

The program XIS is embedded in the Windows®XP 32 and Windows® 7-32 environment and can be handled in the same manner as other standard WINDOWS programs. It allows limited image processing to present the acquired images.
To present acquired images or sequences of images, the program allows the storage of images in various formats. Hard copies of the images can be printed with any installed printer. It is recommended to use a printer allowing resolving 256 grey levels for optimum presentation.

1.3   Liability policy

PerkinElmer endeavours to provide a quality product and takes full responsibility for the correct function of the XIS-Software with its own digital X-Ray detectors. PerkinElmer is not responsible for any misuse of the Software, or for the performance with any detector other than with PerkinElmer detectors. PerkinElmer also takes no responsibility for any hardware or software failures caused by installing and using this software on a third-party computer system. The software is solely intended to operate the PKI digital X-Ray detectors with PKI XRD-FG[X][e] Frame grabbers or PKI digital X-Ray detectors with gigabit ethernet interface for product evaluation purposes. PerkinElmer is not liable for property damage, physical injury, or data corruption. PerkinElmer reserves the right to make changes or modifications to this product without notice.

PerkinElmer Inc. SHALL NOT BE LIABLE FOR DIRECT, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING, BUT NOT LIMITED TO, LOSS OF USE, REVENUE, OR PROSPECTIVE PROFITS RESULTING FROM THE USE OF THIS DOCUMENT OR THE PRODUCT TO WHICH IT RELATES. ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE HEREBY DISCLAIMED.

1.4   Copyright

This software is proprietary to PerkinElmer and dedicated for use in combination with the PKI digital X-Ray detectors with PKI XRD-FG[X][e] Frame grabbers or PKI digital X-Ray detectors with 5GHz WLAN/ gigabit ethernet interface for product evaluation purposes only and without licensing agreement. This document and the product to which it relates are protected by copyright law from unauthorized reproduction.

Notice to U.S. Government End Users
The Software and Documentation are "Commercial Items," as that term is defined at 48 C.F.R. 2.101, consisting of "Commercial Computer Software" and "Commercial Computer Software Documentation," as such terms are used in 48 C.F.R. 12.212 or 48 C.F.R. 227.7202, as applicable. Consistent with 48 C.F.R. 12.212 or 48 C.F.R. 227.7202-1 through 227.7202-4, as applicable, the Commercial Computer Software and Commercial Computer Software Documentation are being licensed to the U.S. Government end users (a) only as Commercial Items and (b) with only those rights as are granted to all other end users pursuant to the terms and conditions herein. Unpublished rights reserved under the copyright laws of the United States.

1.5   Getting Started - The first image

1.5.1   Introduction

This chapter describes the procedures to get the first x-ray images with adequate quality. It explains how one can use correction files with the appropriate settings of the detector integration time and x-ray source parameters.

1.5.2   General considerations

Basically the detector is able to produce images without any correction. These images contain the offset of the readout electronics, the individual offsets of each pixel (dark current) as well as the electronics and pixel gain differences, apart from the x-ray source non-uniformities.
Each column is connected to one channel of the readout electronics with the specific channel offset. This results in a dark image with vertical stripes caused by the individual channel offsets. The dark image may also contain pixels which are brighter than the others caused by a higher dark current. The detector is arranged in groups of multiple readout channels. The groups can deviate in their gain such that one can distinguish blocks of multiple channels in a bright image caused by this gain difference. The panel itself may contain pixels and perhaps row or columns which are defective (totally black or white). To eliminate these detector specific effects and to obtain good quality results each image should be 'offset' and 'gain' corrected and if it is required the defective pixels should also be corrected. The creation of the correction files is described in the chapter 1.6 *How to perform corrections*. The mathematical procedures which are applied to each pixel are described in chapter *1.7 Mathematical description of corrections*.

1.5.3   Connection of the X-Ray Detector

Before starting the connection between X-Ray Detector and host PC please ensure that the frame grabber and the software are installed correctly (chapter 3). If not please begin with the frame grabber and software installation.
In case of a GigE detector (0822/1622) please ensure you have installed a Gigabit Ethernet network card and finally connect the detector to it. When a XRpad detector is used please refer to the XRpad Reference Manuals for connection details.
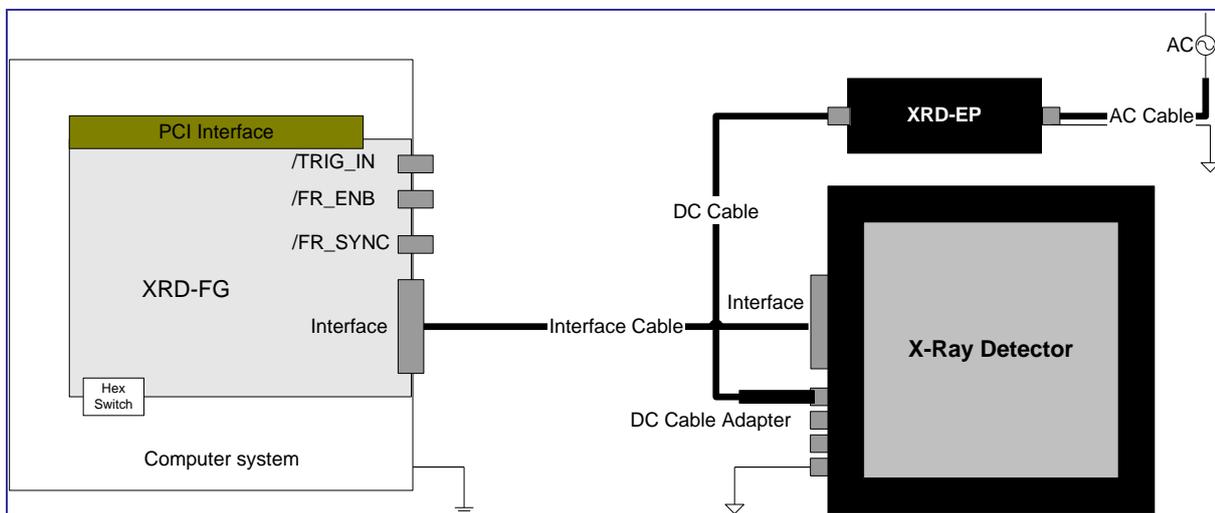
Figure 1: Connections of the XRD 1620

The computer with the frame grabber and the monitor need to be grounded through the protection earth conductors within the power cords. The required potential equalization of the Detector and the XRD power supply has to be managed through the labelled connectors and the potential equalization between both devices is managed through the XRD DC Cable. For safety reasons only original cables and connectors should be used.

The X-Ray detector should be connected as described in the following manner and as shown in Figure 1

1. Shut down the computer
2. Connect the **Detector** and the **power supply** with the potential equalization
3. Connect the **Detector** to the **Frame Grabber** via **Interface Cable** or
   in case of a **GigE Detector,** connect it to the **network adapter** of your host system
4. Connect the **DC Cable** with the Power Connector of the **Detector**
5. Connect the **power supply** and the **DC Cable**
6. Connect the **power supply** with the mains via **AC Cable**.
7. Switch on the **power supply**
8. Switch on the Computer

1.5.4 The first image

Frame-grabber connected Detector	GigE Detector	XRpad Detectors
The detector is powered on and all cable connections are performed. At startup the frame grabber board will be initialized and afterwards a dialog appears to select the mode of the frame grabber board. "Yes" enables the Interrupt Mode, "No" enables the Polling Mode. In both cases the system attempts to initialize the frame grabber board(s) and the connected detectors. The "Cancel" button starts the program without initialization. The initialization can take some time depending on the	Connect the detector directly to the network adapter of the host PC and to the power supply. Then switch on the power supply and the PC. Now wait until the connection between the GigE-Detector and the Host system is established by Windows. When starting the XIS a dialog appears to select the working mode of the system. "Yes" enables the Interrupt Mode, "No" enables the Polling Mode. In both cases the system attempts to initialize the frame grabber board(s) and the	Find all available XRpad detectors by selecting "Enum XRpad". When a detector is found a dialog appears showing the current IP configuration as well as the slected path (LAN / WLAN) for the image transfer. The dialog offers following options: • "Refresh" – update the list • "open detector config" – open the detectors configuration website if the detector IP-settings match the ones from the network adapter[1]. ()

---

[1] the IP address can be also entered directly into the installed web-browser for configuration

number of frame grabber boards and detectors. You can follow up the initialization process via the info box on the lower right corner of the window.	connected detectors. The "Cancel" button starts the program without initialization. The initialization can take some time depending on the number of frame grabber boards and detectors. Alternatively you can setup and configure the detector with "Enum / Setup GbIF Detector". You can follow up the initialization process via the info box on the lower right corner of the window	• "Back" – returns to setup dialog. To initialize the detector and grab images follow the steps in the sections 'Getting the first image' of the XRpad reference manuals. or the steps for the GigE Detectors but use triggered mode in place of Free-Running-Mode

Multiple Detectors:

If more than one detector is connected and initialized a dialog appears that contains a list of all recognized detectors and an active detector has to be selected. All following actions correspond to that active sensor until a different one is selected.

Acquire images:

Now the system is ready to acquire the first image.

The Acquire\Single Shot command acquires a single image. If the detector was not irradiated only a dark image is displayed. The image can be enhanced by the brightness and contrast settings. As explained above the uncorrected dark image contains vertical stripes caused by the electronics offset. By choosing the Continuous acquisition mode the image is refreshed on the display in the selected frame rate.

In the next step the detector should run in the continuous mode and the x-ray source should be switched on to irradiate the detector. The brightness and contrast should be set to default (F2-KEY: 0-65535). If the gray image is displayed the parameters of the x-ray source and detector are in a good range. If a white or a black image is displayed the x-ray source has to be adjusted in the following way:

Displayed image -->	white	black	
Action -->	decrease	increase	–the x-ray tube current
	increase	decrease	–the distance
	append	remove	–additional filters
	decrease	increase	–the x-ray tube voltage

Table 2: X-Ray adjustments

1.6   How to perform corrections

The XRD works as an independent detector to acquire X-ray images. After starting the XIS.EXE software, the detector is automatically initialized and sends out images in its fastest frame time (TIMING-0) among all available ones (Timings menu: list of possible frame times). Please note that XRpad detectors show a blank image in free running mode since it can only be used in triggered mode so an appropriate trigger mode has to be selected before offset and gain images can be acquired.

The X-Ray Detectors (XRD) needs an Offset correction to take into account the dark current of each pixel. In particular, during the warm-up phase of the detector, the Offset is not stable and during this time period the detector use is not recommended. During operation it is recommended to refresh periodically the Offset.

Additionally, a Gain correction is necessary to homogenize differences in pixel sensitivities or to take into account the X-ray beam illumination; therefore it is very important that the whole image area is illuminated homogenously. The Gain correction should be carried out in an optimum dynamic range of the sensor (70-80 % of the full scale range FSR) or in the dynamic range of interest. The radiation intensity used to create the gain image can depend on the application, e.g. if the typical grey level is about 10.000 ADU and the remaining area is saturated, it is recommended

to use a gain image created at 10.000 ADU. The use of an Offset and Gain calibration eliminates offset dependency and therefore any stored Gain correction file can be used for a specific frame time for longer time periods.

The image performance can be enhanced by using the Multiple Gain Correction. For each dynamic range of interest a separate offset corrected and averaged bright image is used as an interpolation point. The maximum number of interpolation points depends on the installed computer memory. It is important that each bright image is completely and homogenously illuminated.

The Pixel correction allows a 'software repair' of defective pixels to enhance image quality. Improper pixel values are replaced with the averaged value of the surrounding eight adjacent pixels where defective pixels are not used. The pixel correction is only performed on specific pixels, mapped in the file PXLMASK.HIS. The delivery package includes the PXLMASK.HIS file for the specific detector. The user can also generate his own correction file. Please be aware that the number of pixels used for the mean correction should be minimized. The pixel correction procedure requires CPU time and depending on its speed, a slower presentation of the acquired images on screen might occur in relation to the selected timing mode of the system. The main screen of the XIS software sends out a warning message displayed on the screen if all of the acquired images are not accepted by the computer.

1.7   Mathematical description of corrections

All correction files are saved in the HIS file format. Offset and pixel correction files use unsigned 16 bit integer to store the pixel data while gain files use unsigned 32 bit integers.

1.7.1   Offset correction

To create the offset correction image an averaged image of a sequence of dark images has to be acquired. These image data are subtracted from the incoming pixel data at acquisition time.

1.7.2   Gain correction

To create the gain image an averaged image of a sequence of Offset-corrected bright images has to be acquired. Afterwards the median value of the pixel data of the whole sensor is evaluated and the entries in the gain image are derived by the following formula

entry = median*65536/(bright_value-offset_value)

For performance reasons the correction is done using integer arithmetic. To improve the precision of the calculation the gain data bits are shifted to left by 16 bits. The gain correction is performed by multiplying the offset corrected pixel data with the gain data and shift the result back.

1.7.3   Multi Gain correction (Gain Sequence)

To create a Sequence for the Multi-Gain correction an offset corrected averaged bright image has to be acquired for each range of interest.
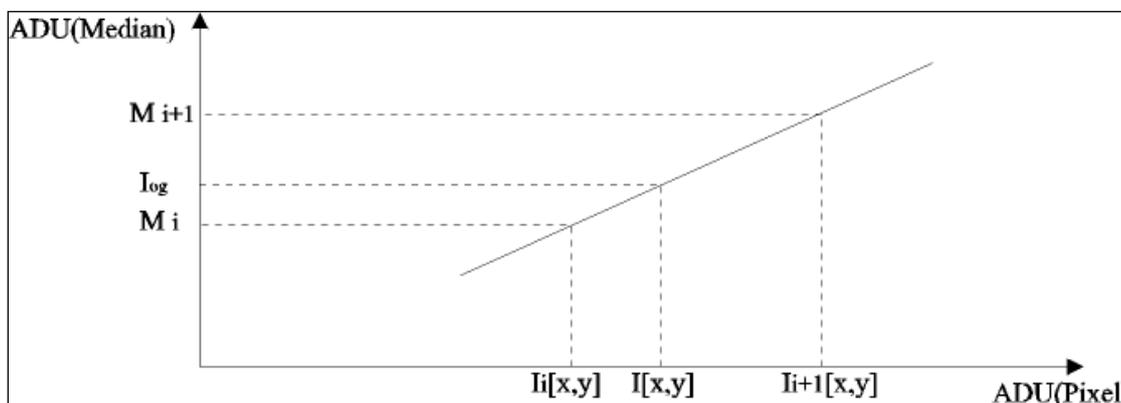
Figure 2: Multi Gain correction

Mi: Median Value of the offset corrected Image i
Ii[x,y]:   Intensity of pixel (x,y) of the offset corrected image i
I[x,y]:   Intensity of pixel (x,y) of the offset corrected input image
Iog[x,y]:   Intensity of pixel (x,y) of the offset-gain corrected output
image

The pixel value Iog[x,y] is calculated with the following formula:

$$I_{og}[x,y] = M_i + \frac{(M_{i+1} - M_i)}{I_{i+1}[x,y] - I_i[x,y]} * (I[x,y] - I_i[x,y])$$

1.7.4   Pixel correction

The pixel correction is done by the XISL function Acquisition_DoPixelCorrection or online when a pointer to a pixel-correction list is passed to Acquisition_AcquireImage. The pixel correction information is stored in a simple HIS file, where the bad pixels are set to a value of 65535 and the others are set to zero. When the pixel correction image is linked to the acquisition unit the XIS derives the correction data from the pixel correction file using the API function Acquisition_CreatePixelMap. For that purpose the pixel marked as defects are corrected by the average of the neighboring good pixels.

1.8 Sorting schemes overview

Depending on the sensor and detector type the data come in different orders from the detector. The XISL sort the data in an internal buffer with highly optimized routines written in machine code. Normally the user does not need to care about sorting because the data returned in the acquisition buffer defined in Acquisition_DefineDestBuffers are in the correct order. If the sensor and detector type is unknown the XIS comes up with a detector type dialog at initialization, where the correct sorting has to be entered. The following detector types and sortings are supported:

XRD 0822, XRD 1622, XRD 1642; XRpad detectors, XRD 1611	0 (no sort)
RID128 *obsolete*	1
RID256 *obsolete*	2
RID128-400 *obsolete*	3
RID1024-100 *obsolete*	4
RID512-400 A0 *obsolete*	5
XRD512-400 A1/A2, XRD 0840	6
XRD512-400 E	7
XRD 1640 A , XRD 0820	8
XRD 1620 AJ	8
XRD 1680 A	9
XRD 1620/21 AM/AN, XRD 1621	11
XRD 16x0 AN CS	12

Table 3: Sorting Schemes

1.9 How to acquire data from several sensors in parallel

This feature of the XIS can be used by plugging in two or more frame grabber boards and connecting them to suitable detectors. But it is not recommended to use different types of frame grabbers together. Each frame grabber has to be set to a unique sensor number at the hex switch (unequal to zero). When using multiple XRpad detectors it is possible to select multiple detectors in the "Enum GbIF detectors"-dialog and press "Init". During the XIS initialization the active sensor has to be selected. Choose Acquire/Continuous to start a continuous data acquisition or acquire correction images or link the correction data before. By the dialog Active Sensor in the Options menu another sensor from the list can be selected. The following actions correspond to the new active sensor. But now new correction files have to be acquired or to be linked for the new sensor. The command Window/Tile shows both acquisition windows.

1.10 How to use function keys

Function keys are available in the acquisition mode for an easy image presentation. The function key **F2** allows the display of the full range of the image, presented in 256 gray levels. The function key **F7** allows to switch to on-line Offset, **F8** to on-line Offset/Gain, **Shift F8** to on-line the Multi-Gain and **F9** is used for on-line Median correction.

F1	HELP	Displays the About Box
F2	DEFAULT	Set to default display mode. Reset brightness and contrast settings.
F3	InfoBox	Displays the initialisation Infobox
F7	OFFSET	Enable/disable Offset correction.
F8	OFFSET/GAIN	Enable/disable Gain/Offset correction.
Shift F8	OFFSET/GAINSEQUENCE	Enable/disable GainSequence/Offset correction.
F9	BAD PIXEL	Enable/disable Pixel correction.
<ESC>	STOP	Stop Acquisition
F11	HEADER	Display of the last acquired HwHeader
Shift F11	Frame Counter	Reset of the detector frame-counter (available for detectors with Header-ID>13

Table 4: Function Keys

1.11 What is new?

➤ Supports the new PCIe Framegrabber Devices **XRD-FGe Opto** and the **XRD-FGe**; both including driver support for Windows XP (32) and Windows7(32/64)[2].

➤ Supports the **XRpad** series devices with Gigabit Ethernet Data Interface and 5GHz WLAN for the Windows Versions Windows7(32/64).

➤ Changed Library-Functions SetAcqData, GetAcqData and GetNextSensor for the 64bit implementation of the XISL-Library

➤ Demo sources (SDK) for 64bit in the SDK64 to show how to use the functions for the 64bit drivers.

➤ Changed Demo sources (SDK) for 32/64bit to show how to use the initialisation functions for the Gigabit Detector interface and for the XRpad.

➤ DriverDeinstaller_x64 for 64bit drivers

➤ Added XRpad Autotrigger support

➤ Bugfix in Acquisition_Close / Acquisition_CloseAll caused by gbif[64].dll which may cause a software crash in case of a lost connection. Available from XIS installation 3-3-2-3.

➤ New drivers for XRD-FGe-Opto 32/64bit allowing it to use the 32bit driver in 64bit environment. Available from XIS installation 3-3-2-4.

➤ Support for XRD 1611 added. Available from XIS installation 3-3-2-4.

---

[2] Please note that the XIS is only running with 32 bit or in Windows XP32 compatibility mode!

2 XRD Hardware Interfaces

2.1 Image acquisition

Following hardware interfaces and operation systems are supported by the SDK:

Interface	OS 32Bit		OS 64Bit
	Windows XP	Windows 7	Windows 7
PerkinElmer XRD FG	X	X	-
PerkinElmer XRD FGe	X	X	X
PerkinElmer XRD FG-X Opto	X	-	-
PerkinElmer XRD FGe Opto	X	X	X
PerkinElmer Gigabit Ethernet	X	X	X
PerkinElmer WLAN	-	X	X

Table 5: Supported hardware interfaces and operation systems

Note: XIS Demo-Software is a 32bit application only

2.1.1 PCI Framegrabber XRD-FG

IB – XRD Interface Bus
1 – /TRIG IN
2 – /FR_EN (Begin of frame)
3 – /FR_SYNC
4 – Status LEDs
5 – HEX Switch

1,2 and 3 operate with TTL signals. Details can be found in chapter 6.1.9.
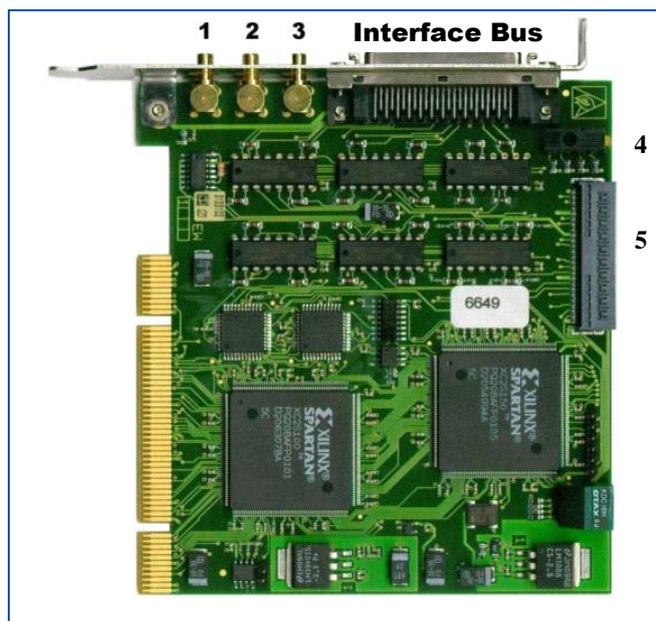
Figure 3: Image of the XRD-FG Frame Grabber

2.1.2 PCI Express Framegrabber XRD-FGe

PCIe –x1

1 – HEX Switch

2 – Status LEDs

3 – XRD Interface Bus

4 – D-Sub connector (details in chapter 6.1.9)
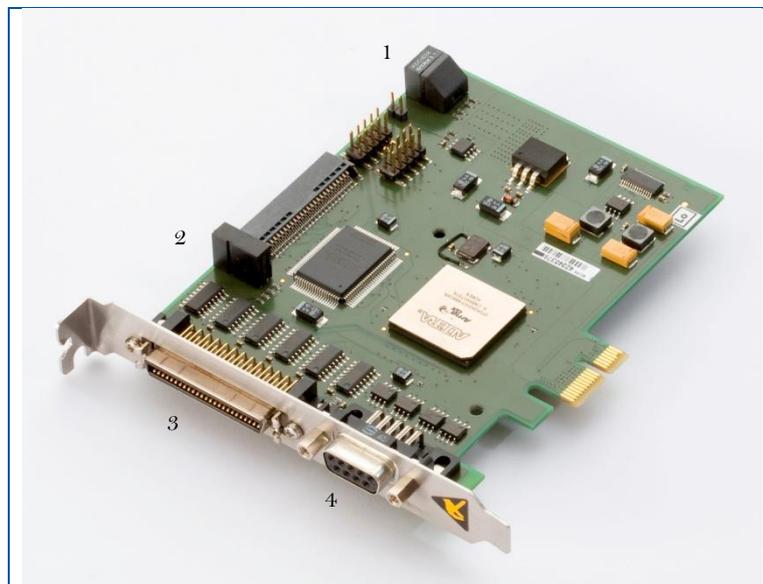
Figure 4: Image of the XRD-FGe Frame Grabber

PCIe –x1

1 – HEX Switch

2 – Status LEDs

2.1.3   PCI-X Framegrabber XRD-FGX Opto

PCI-X Rev 2.0 133MHz 3,3V

1 – HEX Switch

2 – D-Sub connector (details in chapter 6.1.9)

3 – Optical Transceiver Out

4 – Optical Transceiver In
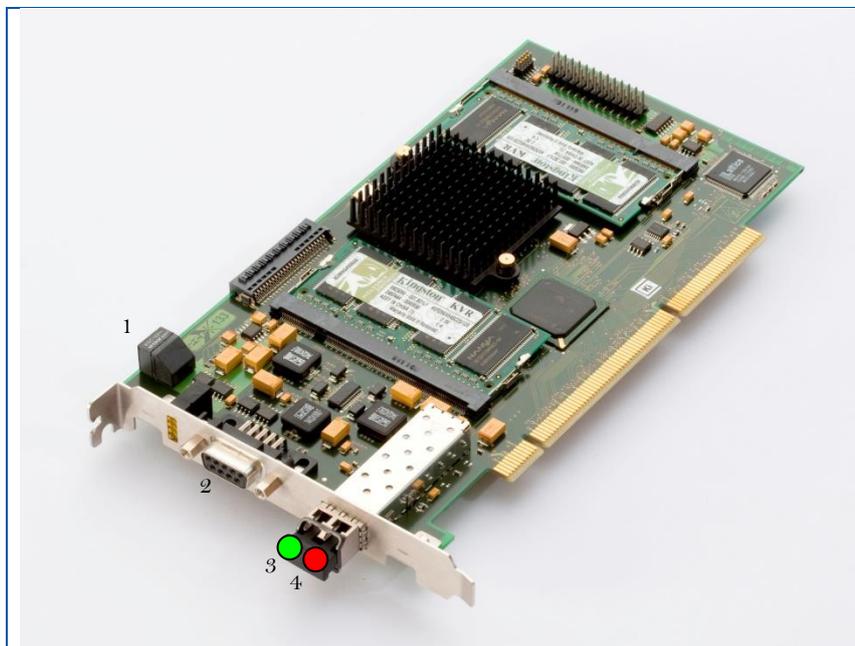
Figure 5: Image of the XRD-FGX Opto Frame Grabber

2.1.4 PCIe Framegrabber XRD-FGe Opto

PCIe –x4

1 – HEX Switch

2 – D-Sub connector (details in chapter 6.1.9)

3 – Optical Transceiver Out

4 – Optical Transceiver In
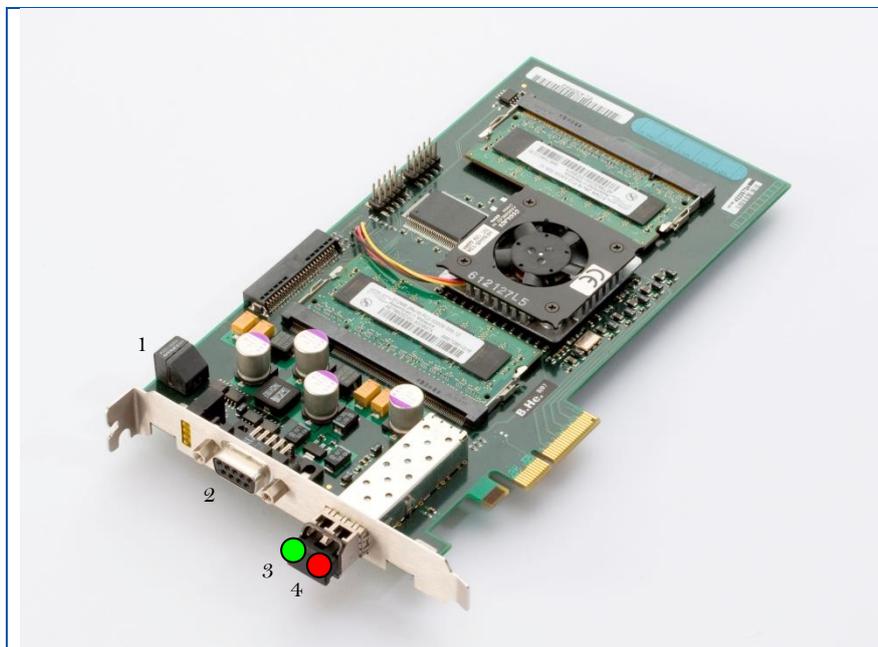
Figure 6: Image of the XRD-FGe Opto Frame Grabber

2.1.5 Features of the Framegrabbers with optical interface

The detector types XRD 1621/41AN and XRD 1611 are equipped with an optical Interface only.
Features are:

➤ Data Rate up to 15fps @ 2k x 2k x 16bit / 4 fps @ 4k x 4k x 16bit
➤ Hardware Image correction (offset/gain/mean) w/o loss of frames up to 10 calibration points for non linear correction.
➤ 5-Level Pixel correction (XRD-FGe Opto only)
➤ Online image rotation for images with image size up to 2k x 2k pixels
➤ Glass fiber Interface (galvanic isolation, robust, detector plug type IP68)
➤ There are two types of the XRD-FGe Opto:
  - 95510216H supports detectors with up to 2k x 2k pixels
  - 95510218H supports detectors with up to 4k x 4k pixels (sdk libraries from installation 3-3-2-4 or higher required)

2.1.6   Gigabit Ethernet Interface

The Gigabit Ethernet Interface is embedded into the detector and is able to interact with standard Gigabit Ethernet network interfaces (RJ45). The recommended way to use this interface is in direct (Point-to-Point) connection to the hosting system to guarantee that no additional network traffic arises for this connection. To connect the Detector with the host device the PKI CAT5e/CAT6 (shielded twisted pair, stranded or solid copper conductor) cable should be used.

Figure 7: Front view of the Gigabit Detector XRD0822

1 – Potential Equalization
2 – Power Input
3 – Trigger Input
4 – Gigabit Ethernet Connector
      Left LED (orange) Traffic
      Right LED (green) indicates Gigabit Link for 2Seconds after Power On
5 – Power LED
6 – Status LED
      OFF – no open software connection
      Flashing green   Free Running mode with active software link
      Flashing amber   Triggered mode with active software link
      Red   Error during Self-Test

2.1.7   XRpad Interfaces

The XRpad can be connected via WLAN or via the Interface and Power Unit XRpad IPU with optional LAN. Please refer to the XRpad Reference Manuals for further details.

3 Installation

3.1 Installation of X-Ray Imaging Software / Installation of the Framegrabber

3.1.1 Framegrabber

To install the frame grabber it is recommended to shut down the computer and unplug the power supply as electricity may cause severe damage to both your motherboard and the frame grabber. In most cases the mainboard has an onboard LED which shows the power OFF mode or the soft-off mode (Power is still on). Hold the grabbers by the edges and try not to touch the chips, leads or connectors. Please place the frame grabber on a grounded antistatic pad whenever the grabbers are separated from the system.

It is recommended to use an exclusive IRQ port, please read your mainboard guide for more information. If more than one frame grabber has to be used in the system the switch on the left side of the grabber has to be set to a unique number for every board.

Please read the readme.txt on the installation CD for the latest information before installing the frame grabber driver and the application software.

1. Shut down the computer
2. Unplug the power supply and remove the computer system's cover
3. Turn the switch of the grabber to a unique number for every board
4. Carefully align the frame grabber's connectors and press firmly
5. Secure the card(s) on the slot with a screw
6. Replace the computer system's cover
7. Restart the computer system
8. Log on to Windows using the administrator account

3.1.2 Gigabit Ethernet / WLAN Interface

If you possess a detector with Gigabit Ethernet / WLAN interface please connect it to a Gigabit network or WLAN interface of your computer system.

In case of a XRpad detector different IP subnet settings for LAN and WLAN are requiered (e.g. use IP: 192.168.2.123 subnet 255.255.255.0 for detector LAN and IP: 192.168.22.123 subnet 255.255.255.0 for WLAN which means that the LAN subnet is 192.168.2. and the WLAN subnet is 192.168.22).

Windows Firewall Settings:

Before a Ethernet Detector can be used you have to ensure that either the Windows Firewall is disabled for the application (XIS.EXE / XISL_Demo.EXE / "Your_App.EXE") or the following ports are excluded from the firewall:

1234 (UDP)	always
3956 (UDP)	always
57635(UDP)	always for XRpad
1000 (UDP)	always (for 1 detector)
1001 (UDP)	always (for 1 detector)
1002 (UDP)	if a second detector is used
1003 (UDP)	if a second detector is used
1004 (UDP)	if a third detector is used
1005 (UDP)	if a third detector is used
…	

Detector IP Setup:

For the Gigabit Detector Setup with XIS please refer to Page: 45 Chapter: Automatic Initialization

3.1.3 Software Installation on Windows®XP and Windows® 7

Before you install the XIS software (32bit only), please make sure either a framegrabber is installed or you use a detector with Gigabit Ethernet interface.

1. Insert the XIS Installation CD-ROM into your CD drive. The installation will start automatically. If not, go to the root directory of the CD and run the executable START.EXE.

2. The XIS SETUP program will lead you through the installation process of the XIS demo software with the option to install the Gigabit Driver and the 32/64bit SDK additionally.

3. Restart the computer system.

4. The XIS is now ready to start.

5. If the initialization of the frame grabber and the detector is successful a corresponding message appears in the status bar.

3.2   Trouble Shooting

XIS: Setup:

Color Resolution:

The setup informs you, that the selected color resolution is different than 256 colors.

Please choose "Settings/Control Panel" in the Windows "Start" menu and double click the icon "Display". Several property sheets appear. Select "Settings" and enter at least "256 color" in the "Color Palette" list box.

XIS and Zooming:

Some computer systems have problems with the XIS zoom functionality in combination with the Microsoft direct draw drivers.

As a workaround you can switch of Direct Draw and Direct3D.

For that please perform the following steps:
-   Open the DOS-Box
-   enter `dxdiag` and press Return (now the direct control panel starts)
-   go to the 3rd tab (display/view)
-   deactivate DirectDraw and Direct3D

XIS: Detector Initialization:

XIS Error 2 // Eltec Error –303

IRQ-Problem

Change the IRQ-Settings in the System Settings Control Panel or in the BIOS. Use another PCI-Slot without IRQ-Sharing or remove not absolutely necessary PCI-boards or use the polling mode.

XIS Error 2 // Eltec Error 1:

The system is unable to initialize Framegrabber. There might be a problem with the software driver

XIS Error 2 // Eltec Error –41:

Virtual device driver not present

No framegrabber driver is loaded or an old driver is still installed in the system but the framegrabber card is already removed .In that case you can use the DriverDeinstaller-Tool to clean up the system.

XIS Error 23:

Hardware header invalid
➤   Check the connections of the detector and the frame grabber board.
➤   Check if there are older libraries of an earlier installation in the path then delete or rename it.
➤   Try the detector setup by the Option/Acquisition dialog
➤   Write down the header information and contact your vendor

Optical Grabber only:

PCI-FGX Opto – PCI Problem

➤   This Message occurs when the PCI-X Slot is only in 32Bit mode or running with 66MHz in place of 100/133MHz. This may be caused by a PCI card connected to a PCI-X slot. So the PCI-X Grabber will not work correct. Please try using another PCI-X slot with 100/133MHz to avoid this sharing problem.

PCI-FGX Opto – Driver Version Problem

➤ This Message occurs when the Library function Pe_TestbusSpeed is not found. Please try to install the latest driver. Be sure that all files of the old driver are removed / uninstalled before. You can use the **new DriverDeInstaller.exe** which is located on the xis-cd to search and remove frame grabber libraries from the system32 directory.
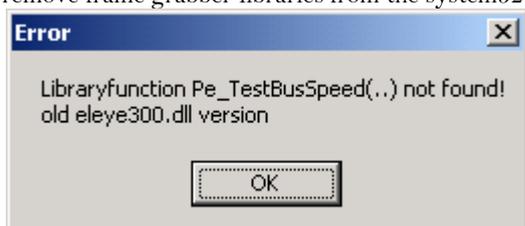
XISL: Image acquisition:

XISL Error 1:

➤ If the function Acquisition_Acquire_Image returns 1, one possible reason can be the memory situation of the host system:

Please check your "boot.ini" file if the flag **/3GB** is set. If this is the case please remove this flag and try to acquire image data after rebooting the system.

However, if the function is not able to allocate enough memory for image acquisition (which at least is the size of the internal, 8 images comprising ring buffer) it also returns this error.

GigE Latency Problems:

➤ Latency issues during image acquisition have been observed using LLA in combination with Windows. It is recommended, to use the DHCP or the Static IP setup for continuous/sequent image acquisitions rather than the LLA solution.

FAQ:

For a selection of frequently asked questions referring to XIS, XISL and frame grabber devices please open the following link:

http://las.perkinelmer.com/downloads/walluf/XIS_XISL_FAQ.pdf

In these FAQs you will also find a nice summary of how to set up a connection to detectors having the Gigabit Ethernet Interface.

3.3 Downloads

The latest common drivers (32Bit and 64Bit) can be downloaded using the following link:
http://las.perkinelmer.com/downloads/walluf/XRD_drivers.zip

The latest common drivers (32Bit and 64Bit) as well as the XIS demo application and API demo source code can be downloaded using the following link:
http://las.perkinelmer.com/downloads/walluf/XIS_Inst_Package.zip

Note: Please contact PerkinElmer applications engineers to verify which software version applies to your detector configuration and how to receive the latest software release!

X-Ray Imaging Software

4   Overview of the Menu Commands

4.1.1   File Menu Commands

The File menu offers the following commands:

New	Creates a new document.
Open	Opens an existing document.
Close	Closes an opened document.
Save	Saves an opened document using the same file name.
Save As	Saves an opened document to a specified file name.
Save Correction files	Saves all loaded correction files.
Import	Imports an existing uncompressed document
Print	Prints a document.
Print Preview	Displays the document on the screen as it would appear printed.
Print Setup	Selects a printer and printer connection.
Exit	Exits XIS.

Table 6: File menu

4.1.2   Edit Menu Commands

The Edit menu offers the following commands:

Copy	Copies data from the document to the clipboard.
Math	A dialog comes up that gives you the possibility to manipulate sensor data by sophisticated build in functions.
Copy Frames	Copies frames from one document to an empty document.
Average	Averages over the images of a sequence.
Select by value	Selects image pixels by their data values.
Select all	Selects all image pixels.
Deselect all	Deselects all image pixels.
Select Rect	Selects a rectangle of pixels by coordinates
Set value	Sets all selected pixels to a specified value.
Create pixel map	All selected pixels are marked as defects and a pixel correction map is created.

Table 7: Edit menu

4.1.3   Acquire Menu Commands

The Acquire menu offers the following commands, which enable the user to acquire images of the detector and to perform suggested corrections to achieve best image quality:

Continuous	Acquires and displays images continuously in the selected timing mode. (Default TIMING0: shortest acquisition time of the detector.)
Single Shot	Acquires a single image.
Sequence	Acquires a sequence of images.
Link Offset Corr	Linking of a stored **Offset** correction file to acquire images corrected with the loaded **Offset** file.
Link Gain Corr	Linking of a stored **Gain** correction file to acquire images corrected with the loaded **Gain** file.
Link Gain Sequence	Linking of a stored **Gain-Sequence** correction file to acquire images corrected with the loaded **Gain-Sequence** file.
Link Pixel Corr	Linking of a stored **Pixel** correction file to acquire images corrected with the loaded **Pixel** file.
Get Offset Image	Acquires a new **Offset** image for later Offset corrected image acquisition.
Get Gain/Offset Image	Acquires a new Offset corrected **Gain** image for later Gain/Offset corrected image acquisition.
Get All Offset Images	Acquires **all Offset** images for later Offset corrected image acquisition. (All frame times available in the Timings menu are automatically acquired)
Build Gain Sequence	Creates a **Gain-Sequence** correction file to be used with the Gain Sequence correction.
Convert to Gain Image	Creates a new gain image from an offset and a bright image. In the bright image a region of interest can be selected to optimize the gain image regarding best presentation of this area.
Set Soft Trigger	Opens a Dialog from which a Single Softtrigger or a repeated SoftTrigger can be send to the Detector.

Table 8: Acquire menu

4.1.4   Detector Menu Commands

The Detector menu offers the following commands:

Mode	Selects the Detector mode. The detector is able to operate the following modes: free running, triggered by external sources triggered by an customized internal timing triggered by software auto triggered (XRpad only)
Timings	Selects the currently active frame time for detector operation.

Table 9: Detector menu

4.1.5   View Menu Commands

The View menu offers the following commands:

Acquisition Bar	Shows or hides the acquisition toolbar
Toolbar	Shows or hides the toolbar.
Status Bar	Shows or hides the status bar.
LUT	Shows or hides the LUT window.
Control	Shows or hides the Control window.
Player	Shows or hides the Player bar (available using Acquire Sequence command).
Zoom Box	Shows or hides the Zoom Box window
Image Data	(obsolete) Shows or hides image data saved to the file
Plot Box	Shows or hides the plot box which displays a graph containing the values of the currently selected image data for each Row, Column or as a histogram.

Table 10 : View menu

4.1.6   Window Menu Commands

The Window menu offers the following commands, which enable arranging multiple views of multiple documents in the application window:

New Window	Creates a new window that views the same document.
Cascade	Arranges windows in an overlapped fashion.
Tile	Arranges windows in non-overlapped tiles.
Arrange Icons	Arranges icons of closed windows.
Window 1, 2, ...	Goes to specified window.

Table 11: Window menu

4.1.7   Options Menu Commands

The Options menu offers the following submenus:

Acquisition	Allows to enter specific acquisition options.
View	Allows to enter specific view options (zooming etc.).
Active Sensor	Allows to change the currently active sensor if more then one sensor is connected to the system.
Sensor	Gives information and allows to set some options for the active sensor

Table 12: Options menu

4.1.8   Help menu commands

The Help menu offers the following commands.

Help Topics	Displays the About Box
About	Displays the version number of this application.

Table 13: Help menu

4.2   File menu

4.2.1   New command

This command creates a new document in XIS. The type of the file can be selected in the New File dialog box.

Toolbar:

Keys: CTRL+N

The following dialog appears if the user creates a new document

Figure 8: New File Dialog

Window name	This name specifies the name of the new document.
rows, columns, frames	This input numbers specify the number of columns, rows and frames of the new document. All data values are set to zero.
type of data	The XIS supports different data types for data acquisition and evaluation. The suitable type could be specified here. All operations done by the XIS are type safe. If a data type isn't suitable for the desired operation a warning is given out and the data type is changed allocating a new suitable data buffer. In this case all old data are lost.

Table 14: Settings of the New File Dialog

4.2.2   Open command

This command opens an existing document in a new window. Multiple documents can opened at once. Use the Window menu to switch among the multiple open documents.

See Window 1, 2, ... command.

Toolbar:

Keys: CTRL+O

4.2.3   Close command

The Close command closes all windows containing the active document. XIS suggests that the changes of document should be saved before closing it. If a document is closed without saving, all changes made since the last time of saving are lost. Before closing an untitled document, XIS displays the Save As dialog box and suggests to name and save the document.
A document can also closed by using the Close icon on the document's window.

4.2.4   Print command

This command prints a document. First a Print dialog box appears to specify the range of pages to be printed, the number of copies, the destination printer, and other printer setup options.

Toolbar:

Keys: CTRL+P

The following options can be used:

Printer	This is the active printer and printer connection.	
Setup	Displays a Print Setup dialog box to change the printer and printer connection.	
Print Range	Specify the pages you want to print:	
	All	Prints the entire document.
	Selection	Prints the currently selected text.
	Pages	Prints the range of pages you specify in the From and To boxes.
Copies	Specify the number of copies you want to print for the above page range.	
Collate Copies	Prints copies in page number order, instead of separated multiple copies of each page.	
Print Quality	Select the quality of the printing. Generally, lower quality printing takes less time to produce.	

Table 15: Settings of the Print Dialog

4.2.5   Print Preview

This command displays the active document as it would appear when printed. When this command is called, the main window will be replaced with a print preview window in which one or two pages will be displayed in their printed format. The print preview toolbar offers you the following options:

Print	Bring up the print dialog box, to start a print job.
Next Page	Preview the next printed page.
Prev Page	Preview the previous printed page.
One Page / Two Page	Preview one or two printed pages at a time.
Zoom In	Take a closer look at the printed page.
Zoom Out	Take a larger look at the printed page.
Close	Return from print preview to the editing window.

Table 16: Settings of the Print Preview Dialog

4.2.6 Print Setup

This command presents a Print Setup dialog box to specify the printer and its connection.

4.2.7 Save command

The Save command saves the active document to its current name and directory. Saving a document for the first time, XIS displays the Save As dialog box to name the document. To change the name and directory of an existing document before saving, choose the Save As command.

Toolbar:

Keys: CTRL+S

4.2.8 Save As command

Use this command to save and name the active document. XIS displays the Save As dialog box to name the document.

To save a document with its existing name and directory, use the Save command.

4.2.9 Save Correction files

Use this command to save all currently loaded correction files. The files are stored in the correction folder defined in the Options/Acquisition dialog.

4.2.10 Exit command

Use this command to end the XIS session. The same effect has the Close command on the application Control menu. XIS prompts to save documents with unsaved changes.

Mouse: Double-click the application's Control menu button.

Keys: ALT+F4

4.2.11 Import command

This command opens an existing file containing raw data in a new window. In the appearing dialog the number of columns, rows, frames and the data type can be selected.

4.2.12 1, 2, 3, 4 command

Use the numbers and filenames listed at the bottom of the File menu to open the last four documents you closed. Choose the number that corresponds with the document you want to open.

4.3   Edit menu

4.3.1   Edit build sequence

This menu entry opens the "Build Sequence" dialog to edit available sequences of images by inserting or appending data from other images or to create a new sequence from existing images. The "Build Sequence" dialog looks like the following image:

Figure 9: Build Sequence Dialog

The available group box shows all available drives, directories and files. The currently loaded images are listed under the drive "computer" in the drives list box. To list stored images click on the items of the drive box and the entries of the above tree control that displays the directory structure.

Sort by radio buttons	Change the sorting of the available images by the corresponding radio buttons
append to sequence	Check this radio button to append images to a sequence.
insert at button	Check this radio button to insert images at a specified location in the data stream. The last frame of the sequence is specified by the "insert at" edit box.
Sequence edit box	Drag and drop a sequence file from the "available data" box or edit the name. If the file name isn't loaded and not available on the storage media a new data window is created and the data are appended.
To use to build sequence box	This list contains the files which will appended to or inserted into the sequence. Drag and drop files from the "available data" list box.

Table 17: Settings of the Sequence Dialog

4.3.2   Edit Math

The Edit Math menu entry opens the "enter expression" dialog box. The build in parser supports several mathematical image calculations, e.g. addition of image data to numbers or other image data, the subtraction of images and numbers, the division and multiplication of images and numbers and averaging of different images and numbers.
The status of the parser is shown in the Status Bar.
The details of the parser are described in the chapter "Mathematical expressions".

4.3.3 Copy

This command copies selected data into the clipboard. If no data selected, the Copy command is unavailable. Copying data to the clipboard replaces the previously stored data.

Toolbar:

Keys: CTRL+C

4.3.4 Copy Frames

This command copies a number of frames into a new document. In the appearing dialog the range of frames can be specified.

4.3.5 Average

This command derives the average of the frames provided in the active document and copy the result into a new empty document.

4.3.6 Select by Value

This command selects all pixels of the active document that are within or out of a specified range. The selected pixels appear on the screen in the manner, which is adjusted by the control box. All following actions are related to this selection.
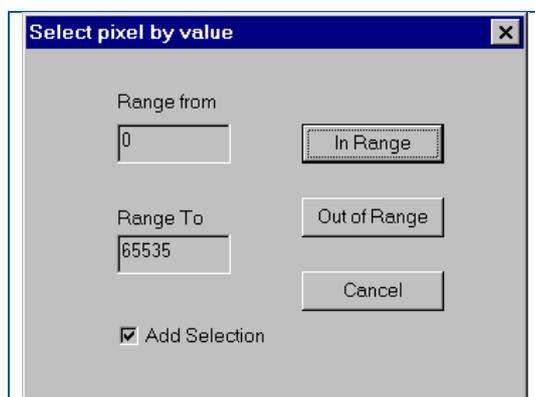
Figure 10: Select Pixel by Value Dialog

Range from	This edit control specifies the lower limit of the selection range.
Range to	This edit control specifies the upper limit of the selection range.
Add Selection	By this button the new selection is added to an older one.
In Range	By this button all pixels within the specified range are selected.
Out of Range	By this button all pixels out of the specified range are selected.
Cancel	This button aborts the selection process.

Table 18: Settings of the Select Pixel by Value Dialog

4.3.7 Select All

This command selects all pixels of the active document. The selected pixels appear on the screen in the manner, which is adjusted by the control box. All following actions are related to this selection.

4.3.8 Deselect All

This command deselects all pixels of the active document.

4.3.9 Set Value

The Set Value command allows the change of pixel values manual. This function is useful for editing pixel maps (see create pixel map).

After selecting this menu entry the following dialog appears:
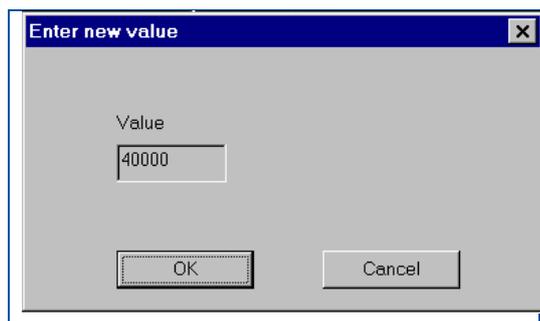
Figure 11: Enter new Value Dialog

Value	Enter the new value of the pixels.
Ok	Press this button to accept the new value.
Cancel	This button aborts the selection process.

4.3.10 Select Rect

Selects a rectangle of pixels by coordinates

4.3.11 Create Pixel Map

This command creates a pixel map from a data file.

A pixel map is a normal data document that can be used for a pixel correction. Defective pixels are distinguished from working pixel by their values in the pixel map. Defective pixels have values of 65535, the values of working pixels are different from this value.

This menu entry creates an empty data document and all selected pixels of the source document are marked as defect pixels in the new created pixel map.

4.4   Acquire menu

4.4.1   Single Shot

One single image of the detector at the selected frame time (see Timings menu) is acquired by the Single Shot command. The image buffer contains the data of the acquired image.

The Function keys are not available during single shot acquisition, only the ESCAPE key can be pressed to interrupt this action. The Image corrections are Offline available (Offset, Gain or Pixel correction) for details see the chapter "image corrections". Before the next shot a dialog appears to choose the manner of further processing of the actual image frame (see also Overwrite Data). If the next data's are acquired into a new window, the last actual correction setting is performed during the next image acquisition cycle. To change the current correction setting the Windows menu can be used to close the correction files or the Acquire menu to link other files.

Toolbar:

4.4.2   Sequence Command

By this command a defined image sequence at the selected frame time (see Timings menu) is acquired. The image buffer contains the data of all specified acquired images. The Function keys are not available during sequence acquisition, only the ESCAPE key can be pressed to interrupt this action. The image corrections (Offset, Gain or Pixel correction) can only be enabled or disables before starting the sequence. If the image corrections are disabled it is possible to use the corrections Offline, for details see the chapter "image corrections". The sequence mode offers different options to acquire a sequence of images, which can be set in the dialog below:

- Seq. of averaged Frms: A sequence of averaged frames is acquired. In the example below a sequence is created consisting of 30 images, each of them is an average of 4 subsequently acquired images.
  This function is only available for XRD-FGe OPTO and XRD-FGX OPTO. The amout of Frames to average must be 2 to the power of n.
- Seq. one buffer: Acquire a sequence of n images. If desired, a number of *skipframes* frames can be skipped before each acquired one.
- Average: Acquire a sequence of n images and create an average image as result. In case of the XRD-FGe OPTO and the XRD-FGX OPTO the averaging is done onboard when the amount of Frames to average is 2 to the power of n.
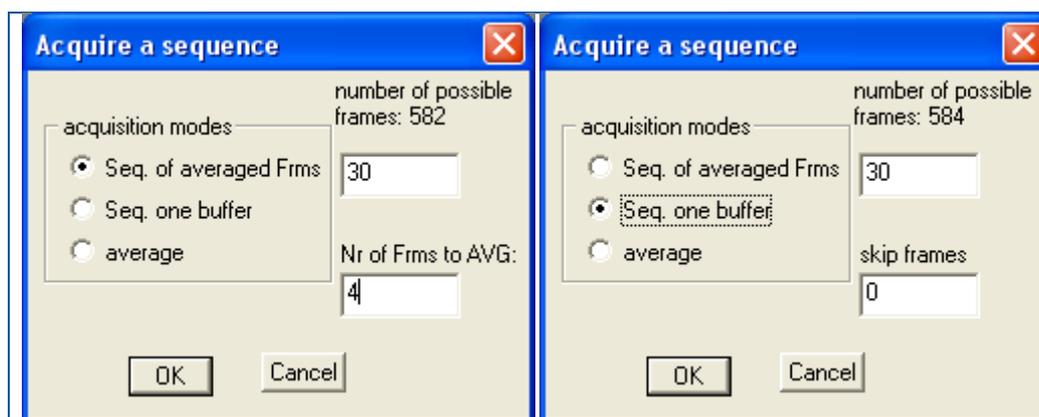
Figure 12: Acquire Sequence Dialog (Seq. of averaged Frms / Seq. one buffer)

Additionally: If there has already been data acquired but not stored, a dialog pops up whether to override the old data or acquire into a new window.

To change the current correction settings the Windows menu can be used to close the correction files or the Acquire menu to link other files.

Toolbar:

Keys: CTRL+SHIFT+S
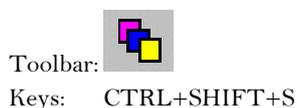
4.4.3　　　　Continuous Command

The **Continuous** command acquires continuously images of the detector at the selected frame time (see Timings menu). The image buffer contains always the data's of the last acquired image cycle. The next acquired image clears the current frame buffer and overwrites the old image.

The call of the **Continuous** command appears in a dialog where the number of images per cycle can be edit. The number of frames per cycle can be selected between one and the displayed maximum number of possible frames, which are depending of RAM and open windows. For memory reasons the default number is recommended. The number of skipped frames can also be edit in the dialog. With the Function keys the online corrections can be enabled or disabled (Offset, Gain or Pixel correction). The image presentation can be changed using the View menu. To stop the continuous acquisition the user can press the ESCAPE key or call **End Acquisition**. The last acquired image is displayed and can be further processed. If more than one frame is in the buffer images of this cycle can be selected by the **Player** command in the View menu. The image presentation can be changed using the View Options.

Before next acquiring of images a dialog appears to choose the manner of further processing of the actual sequence (see also Overwrite Data). If the next data's are acquired into a new window, the last actual correction setting is performed during the next image acquisition cycle. To change the current correction setting the Windows menu can be used to close the correction files or the Acquire menu to link other files.
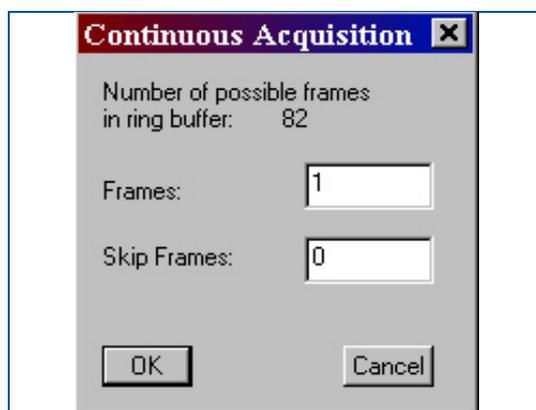
Figure 13: Continuous Acquisition Dialog

Toolbar:

Keys:　　CTRL+SHIFT+C

4.4.4　　　　Get Offset Image

The **Get Offset Image** command generates a new Offset Image file of the detector at the selected frame time (see Timings menu). The number of frames to average for the Offset image can be edit in the appearing dialog. The new correction image is used to perform later the correction of the new acquired images. Please be aware, that a corrected image should not be corrected twice.
See also: Use of the Offset Correction.

The Offset image is automatically linked and used for the next acquired images. To switch between correction and non correction of a running acquisition the Function key (F7) can be used. To stop generally the image correction close the linked correction files by the Window menu commands.

Keys:　　CTRL+SHIFT+O

4.4.5 Get Gain/Offset Image

The **Get Gain/Offset Image** command generates a new Gain-Offset Image file of the detector at the selected frame time (see Timings menu). The number of frames to average for the new Gain image can be edit in the appearing dialog. The new correction image is used to perform later the correction on the new acquired images. Please be aware, that a corrected image should not be corrected twice. See also: Use of the Gain/Offset Correction.

The Gain/Offset image is automatically linked and used for the next acquired images. To switch between correction and non correction of a running acquisition the Function key (F8) can be used. To stop generally the image correction close the linked correction files by the Window menu commands.

Keys: CTRL+SHIFT+G

4.4.6 Get All Offset Images

This command generates a set of new Offset Image files of the detector at **all** available frame times. The number of frames to average for the new Offset image files can be edit in the appearing dialog. The new correction image files are used to perform later the correction on the new acquired images. Please be aware, that a corrected image should not be corrected twice.

The Offset image files are automatically linked concerning their frame time and used for the next acquired images. To switch between correction and non correction of a running acquisition the Function keys can be used. To stop generally the image correction close the linked correction files by the Window menu commands.

4.4.7 Link Offset Correction

The **Link Offset Correction** command loads defined Offset correction files of the detector for the selected frame time (see Timings menu). The linked file is used to perform directly the correction for the actual frame. Please be aware, that a corrected image should not be corrected again. The file is selected by the File Selection dialog box. To acquire Offset correction files see Get Offset Image. This file is also used for the next acquired images. To switch between correction and non correction of a running acquisition the Function key (F7) can be used. To stop generally the image correction close the linked correction files by the Window menu commands.

4.4.8 Link Gain Correction

The **Link Gain Correction** command loads defined Gain/Offset correction files of the detector for the selected frame time (see Timings menu). The linked file is used to perform directly the correction for the actual frame. Please be aware, that a corrected image should not be corrected again. The file is selected by the File Selection dialog box. To acquire Gain/Offset correction files see Get Gain/Offset Image. This file is also used for the next acquired images. To switch between correction and non correction of a running acquisition the Function key (F8) can be used. To stop generally the image correction close the linked correction files by the Window menu commands.

4.4.9 Link Gain Sequence Correction

The Link Gain Sequence Correction command loads defined GainSequence/Offset correction files of the detector for the selected frame time (see Timings menu). The linked file is used to perform directly the correction for the actual frame. Please be aware, that a corrected image should not be corrected again. The file is selected by the File Selection dialog box. To acquire Offset correction files see Get Offset Image. The Gain Sequence correction file must be created with the command Acquire Build GainSequence command. This file is also used for the next acquired images. To switch between correction and non correction of the acquired images the Function Key (Shift+F8) can be used or to stop generally the correction close the linked correction files by using the Window menu commands.

4.4.10 Link Pixel Correction

The Link Pixel Correction command loads a defined pixel correction file of the detector. The linked file is used to perform directly the correction for the actual frame. Please be aware, that a corrected image should not be corrected again. The file is selected by the File Selection dialog box. This file is also used for the next acquired images. To switch between correction and non correction of a running acquisition the Function key (F9) can be used. To stop generally the image correction close the linked correction files by the Window menu commands.

4.4.11 Acquire Build GainSequence

This menu entry opens the "Build Sequence" dialog to create a sequence of offset-corrected bright images to be used with the GainSequence-correction. The selected images will be sorted by median. Beware that the XRD-FGX/FGe OPTO can handle a sequence with up to 10Frames. The "Build Sequence" dialog looks like the following image
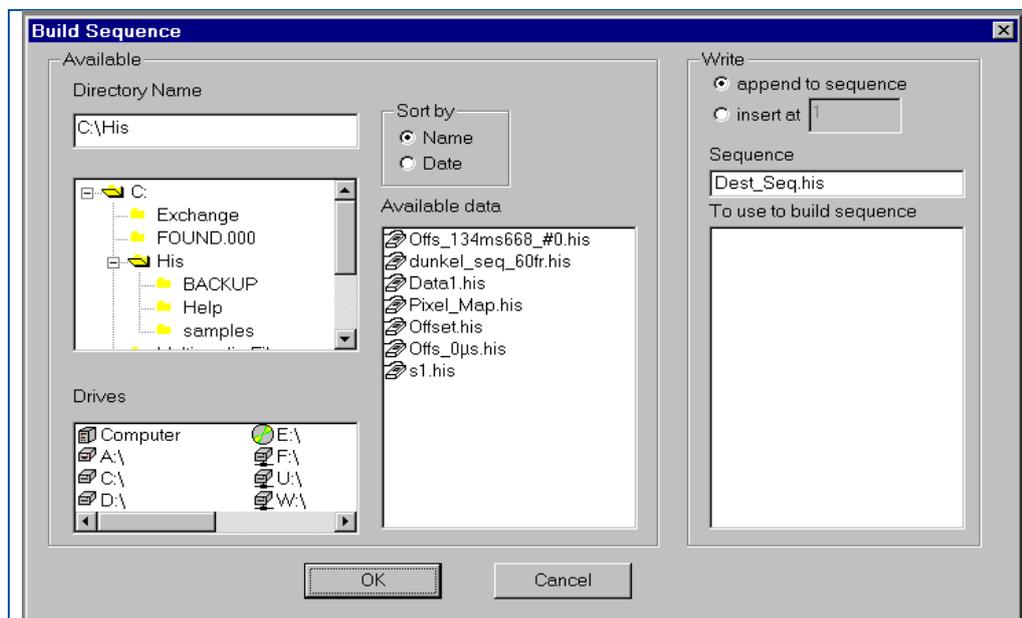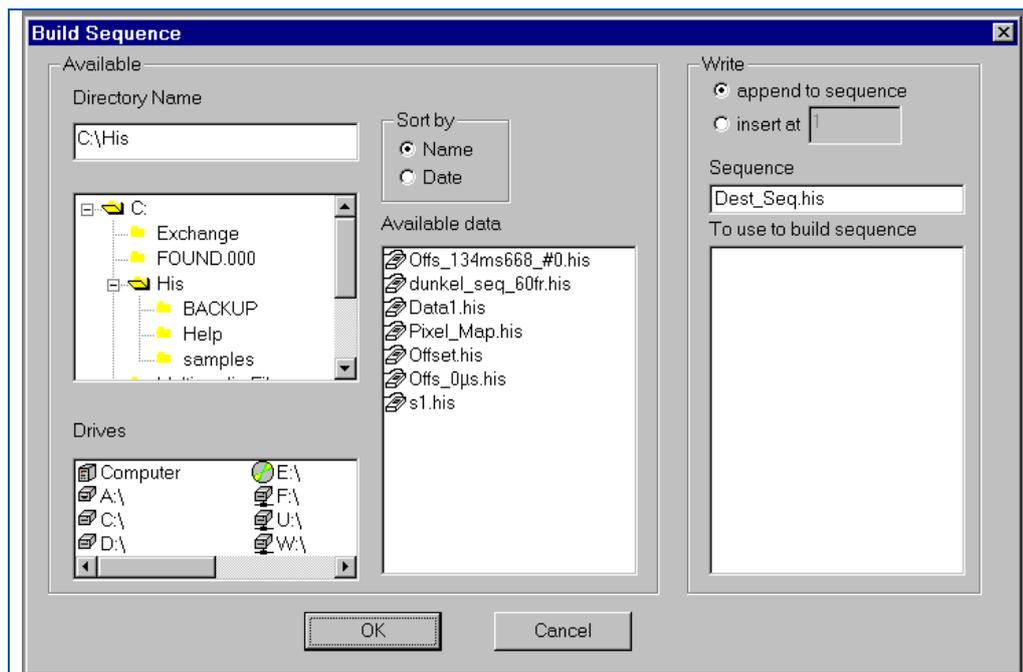

Figure 14: Build GainSequence Dialog

The available group box shows all available drives, directories and files. The currently loaded images are listed under the drive "computer" in the drives list box. To list stored images click on the items of the drive box and the entries of the above tree control that displays the directory structure.

The "write" group provides all controls regarding the destination sequence.

Sort by radio buttons	Change the sorting of the available images by the corresponding radio buttons
append to sequence	Ignored!
insert at button	Ignored!
Sequence edit box	Drag and drop a sequence file from the "available data" box or edit the name. If the file name isn't loaded and not available on the storage media a new data window is created and the data are appended.
To use to build sequence box	This list contains the files which will be appended to or inserted into the sequence. Drag and drop files from the "available data" list box.

Press the Ok-Button to create a Sequence of the selected files sorted by Median.

4.4.12 Acquire Convert to Gain Image

This function creates a new gain image from a dark image file and a bright image file. Compared to the automatic creation of the gain image by Get Gain/Offset Image this routine has the advantage that the user can select a region of interest in the bright image to optimize the gain image regarding best presentation of data in this area. Afterwards the new gain image has to be linked by Link Gain Correction.

The call of Convert to Gain Image comes to a dialog where the dark, the bright and the new gain image has to be named. The available data files can be shown by selecting one edit box and pressing the insert key on the keyboard. In the appearing File Selection dialog the files can be found and entered.

4.4.13 Acquire End Acquisition

This command stops the continuous acquisition. The last acquired image is displayed and can be further processed.

Keys: Esc

4.4.14 Acquire Set Soft Trigger

This command opens a Dialog from which a Single Softtrigger or a repeated SoftTrigger can be send to the Detector when the detector is in Softtrigger-Mode.

4.5 Detector

4.5.1 Timings menu

The Timings menu enables the setting of different frame times for the image acquisition of the detector. Eight different frame times are available. TIMING0 is the shortest possible frame time of the specific detector. The detector starts automatically in the first timing.

Example of the default timings menu for the XRD/RID 512-400 A:

(TIMING 0)	134 ms	(Shortest available frame time of the detector.)
(TIMING 1)	200 ms	
(TIMING 2)	400 ms	
(TIMING 3)	800 ms	
(TIMING 4)	1600 ms	
(TIMING 5)	3200 ms	
(TIMING 6)	6400 ms	
(TIMING 7)	12800 ms	

Table 19: Displaying different Timings

4.5.2 Detector Mode

Four different acquisition modes are available. They are called "free running", "external triggered", "internal triggered" and "soft triggered".

➤ The free running mode means that the detector sends out continuously frames according to the selected frame time. This is the default mode.

➤ The external triggered mode means that the detector sends a frame after triggering by an external pulse and ignores all other incoming trigger pulses until the selected frame time has elapsed. After that the detector can be triggered by a new pulse.

➤ The internal triggered mode means that each frame time between fastest timing and 5 seconds can be selected and the frame grabber triggers the detector by this frame time.

➤ The soft triggered mode means that each frame time between fastest timing and 5 seconds can be realised by sending a trigger signal via the software API or the Menu Item Acquire=>Set Softtrigger.

Note: All three trigger sources can be used with the trigger modes framewise, start/stop and DDD which can be selected by the detector option diaolog (chapter 4.8.5)

4.6 View menu

4.6.1 Status Bar

The left area of the status bar shows online information for the menu or Tool Bar entry where the cursor is above or shows the status of an executed command. A check mark appears next to the menu item when the Status Bar is displayed.
For details see the chapter Status Bar.

4.6.2 Toolbar

The Toolbar command displays and hides the Toolbar, which includes buttons for some of the most common commands in XIS, such as File Open. A check mark appears next to the menu item when the Toolbar is displayed.
For details see the chapter Toolbar

4.6.3 Acquisition Bar

The Acquisition Toolbar command displays and hides the Acquisition Toolbar, which includes buttons for some of the most special commands in XIS, such as Acquisition. A check mark appears next to the menu item when the Toolbar is displayed.
For details see the chapter Acquisition Toolbar

4.6.4 LUT – Look-Up-Table

Hides or Shows the LUT – Look-Up-Table.

The Lock-Up-Table represents the currently selected LUT range in a graphic bar within 256 gray levels. The lowest intensity is represented black, the brightest intensity white. The fading uses the 16 bit range for the actual values.

NOTE:

Exceptionally the values for Gain/Offset correction images are represented in a 32 bit integer mode.

4.6.5 Player

This command is only available if sequences are loaded or acquired.

To show the images and to select a specific frame of the sequence the **Next** and **Previous** button are used. In contrast to the step by step representation the **Play** button starts a continuous playing of all images of the sequence.

Toolbar:

4.6.6 ZoomBox

The ZoomBox is a utility to zoom into the area located at respective the cursor position. The zoom can be intensified using the mouse wheel[3]. Starting at a size of 101*101 pixels being up-scaled, the zoom level can be changed up to 5*5 pixels maximum zoom.

Below the section showing the scaled pixels the ZoomBox provides other features.

Considering image data these are

Position	Current cursor position.
Value	Pixel value at the cursor position.
Window Size	Size n^2 of the currently zoomed section.
Median	Median of the currently zoomed n^2 section.
Sigma	Sigma of the currently zoomed n^2 section.

In case the document considered is a fault mask provided values are:

Position	Current cursor position.
Value	Fault which is set at the current cursor position.
Window Size	Size n^2 of the currently zoomed section.
Fault Density	Fault Densitiy of the currently zoomed n^2 pixels (which is the ratio of good and fault pixels within the n^2 window).
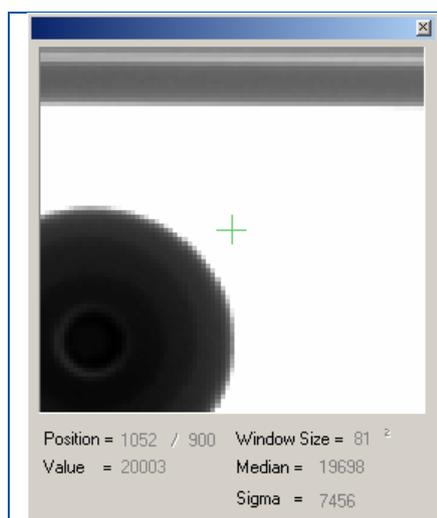
Figure 15: Zoom Box

The ZoomBox can also be activated by clicking the Toolbar button

---

[3] During usage of the ZoomBox the mouse wheel functionality for image scrolling is deactivated. The image can still be scrolled by moving the scroll bar at the window rim though.

4.6.7 View Control Box

The View Control Box appears automatically when an image is acquired opened or imported. To hide and recall the Control Box the menu entry **View** in the View menu can be used.

The detector acquires images of 16Bit, but Windows based computers can only display 8Bit. By the View Control Box the interested grey levels can be selected and displayed on the screen. The selection can be done automatically or manually (zooming).

The Control Box contains the following features:

Brightness	Select brightness for image presentation.
Contrast	Select contrast for image presentation.
LUT Range	Select **Bright** and **Dark** values for image presentation.
Full Range	This box represents always the minimum (0) and maximum (65535) values in 16 bit range.
Invert	Inverts the actual image presentation.
Track Range	Automatic tracking within a region of interest based on evaluated minimum and maximum value of this region. It is recommendable to use this function in the Continuous Acquisition mode for viewing interesting sample parts under X-ray illumination.
Zoom	If this radio button is checked the program evaluates the minimum and maximum pixel values of the selected region of interest and presents this value range in 256 gray levels.
Equalize	Performs image enhancement by equalizing all gray levels in a region of interest.
Full	Allows jumping into the full presentation mode of the LUT Range.
Standard	Selection type Standard
Diagonal	Selection type Diagonal
X1 / y1	Upper left coordinates (absolute) of Standard Selection
X1 / y1	Lower right coordinates (absolute) of Standard Selection
Add Selection	If checked, add Standard selection to existing one. Else delete existing one.

Table 20: Zoom settings of the Control Box
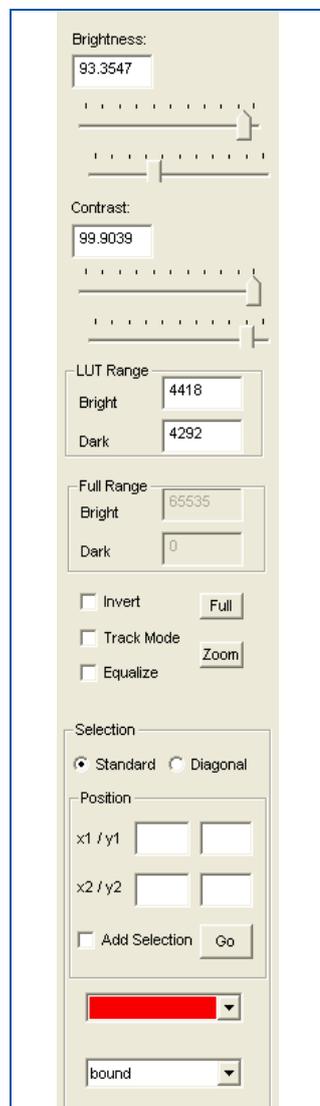
Figure 16: Control Box

To mark **the region of interest** within the image the mouse pointer has to be moved to the starting point and the left button has to be pressed while moving to the end point.

By moving from the upper left to a desired lower right point, the user selects a rectangular shaped window, presented in inverted values.

By moving from lower left to upper right a horizontal line is selected.

By moving from upper right to lower left, a vertical line is selected.

The color and the mode presenting the region of interest can be changed in the listbox on the bottom of the color control window. The different modes are:

Fill	The region of interest is filled with the selected color, it is default
Bound	The region of interest is bounded by the selected color.
And	The selected color will be combined with the corresponding pixel values by a logical and operation within the region of interest.
Or	The selected color will be combined with the corresponding pixel values by a logical or operation within the region of interest.
Hide	Region of interest is hidden.

Table 21: ROI settings of the Control Box

4.6.8   View Image Data

Shows or hides image data previously saved to the file. This function is obsolete

4.6.9   View Plot Box

The Plot Box is a utility to Show a Graph for the currently selected image data for each row, column or as a histogram. It will be updated when a new area is selected or a new frame has been acquired.
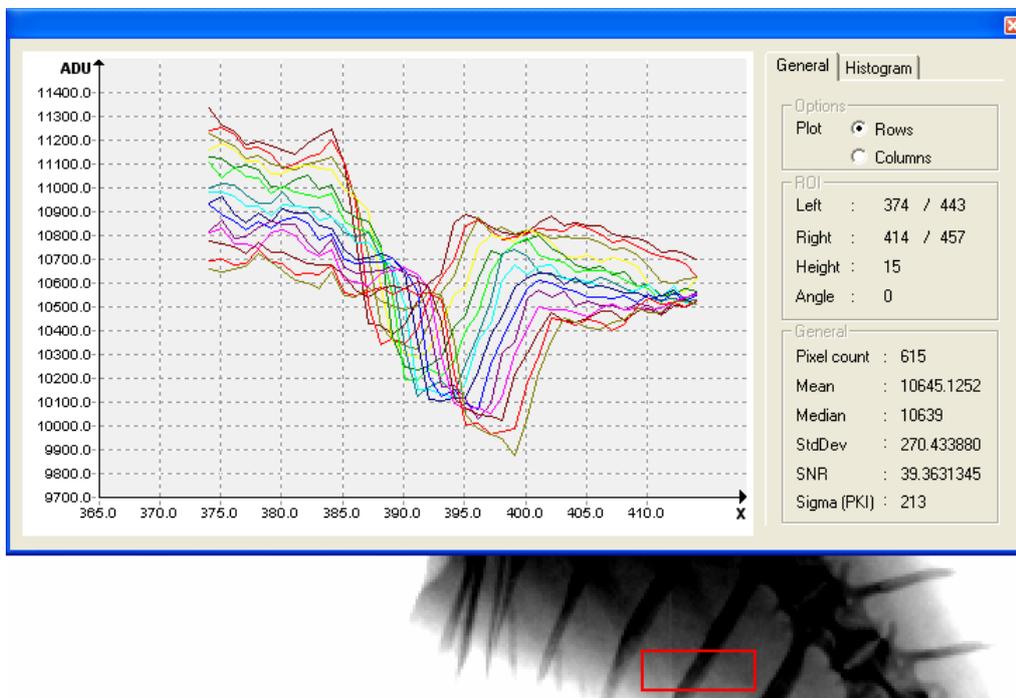
Figure 17: Plot Box – Plot Rows
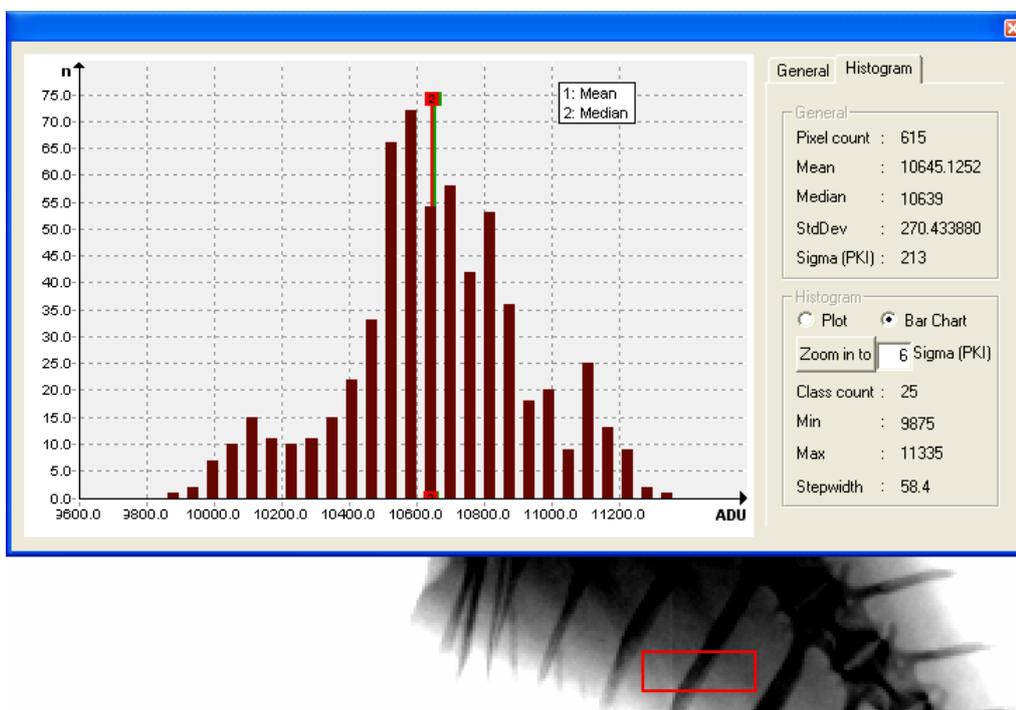
Figure 18: Plot Box – Plot Histogram

4.7   Window menu

4.7.1   New Window

The New Window command opens a new window with the same contents as the active window. Multiple document windows can displayed in different parts or views of a document at the same time. If the contents in one window is changed all other windows containing the same document reflect those changes. When a new window is created it becomes to the active window and is displayed on top of all other open windows.

4.7.2   Cascade

This command arranges multiple opened windows in an overlapped fashion.

4.7.3   Tile Horizontal

This command arranges vertically multiple opened windows in a non-overlapped fashion.

4.7.4   Tile Vertical

This command arranges multiple opened windows side by side.

4.7.5   Window Arrange Icons

This command arranges the icons for minimized windows at the bottom of the main window. If there is an open document window at the bottom of the main window, some or all of the icons displayed below this document window and are not visible.

4.7.6   1, 2, ... command

XIS displays a list of open document windows at the bottom of the Window menu. A check mark appears in front of the document name of the active window. Choose a document from this list to make its window active.

REFERENCE MANUAL

4.8   Options menu

4.8.1   Acquisition

This command restarts the sensor initialization and allows selecting the initialization type:
"yes" for automatic initialization, "no" for manual setup or "Enum / Select GbIF Detector" to
initialize/configure a GigE Detector. The "Cancel" button starts the XIS without initialization of
the detector.

4.8.1.1   Automatic Initialization

The selection of automatic initialization leads to a dialog asking for the mode in which the frame
grabber will run.
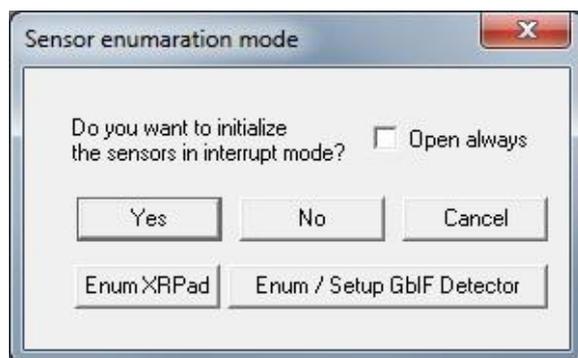
Figure 19: Initialization dialog

Select "**yes**" to run the hard- and software in Interrupt Mode and "**No**" to use the Polling Mode.
If the "Open always" radio button is checked the XIS opens the requested communication channel
regardless if it has been already captured by another process running on the system. It isn't
recommended to use this option except for debugging because the XISL can't free all resources in
one process that were allocated by another process.
When automatic initialization is choosen XIS scans the PCI[X][e] bus for plugged in frame
grabbers and tries to detect connected detectors. If it recognizes more than one detector it asks
you to select a default one. The initialization process is ready if the message "initialization
successful" appears in the status bar.
Note: This function also tries to open Gigabit Ethernet Detectors if the Standard Gateway of the
Detector is zero. This is for example the case when the detector is connected Point to Point per
LLA (detector default) or when a static IP is used without a defined Standard Gateway.

If "**Enum / Select GbIF Detector**" is choosen the software searches for detectors with ethernet
interface that are connected to the computer via Point to Point or the LAN and displays the
results in the Dialog "Enum GbIF Detectors" which is described below.

"Enum XRpad" sends a network broadcast through all avaible network adapters and provides a
list of recognized XRpad detectors which can be initialized with "Enum GbIF Detectors"

4.8.1.2 Enum GbIF Detectors

This dialog shows all Gigabit Ethernet Detectors connected to the system via Point to Point or LAN. This list can be retrieved using the functions **Acquisition_GbIF_GetDeviceCnt** in combination with **Acquisition_GbIF_GetDeviceList**. In case a detector is connected but not listed please check whether your firewall does not block the required ports and the connection has finished establishing (refer to chapter 3.1.2 Gigabit Ethernet / WLAN Interface. XRpad detectors connected via LAN and WLAN will also be listed in this dialog.
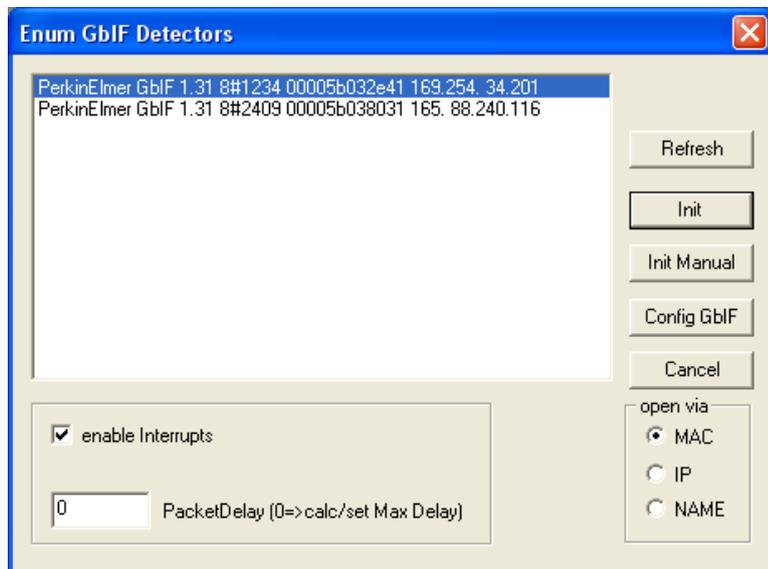
Figure 20: List of found detectors shown by Enum GbIF Sensors dialog

Refresh:

Update the detector list.

Init:

Retrieve the detector settings and initialize the selected detector.

Init Manual:

Initialize the selected detector but enter the detector size and sorting manualy like described in Chapter

Manual Setup below.

Config GbIF:
Configure the Ethernet IP setting for the selected Detector like described in Chapter **Configure GbIF** below.

Cancel:
Close this Dialog without initialization a GbIF Detector.

enable interrupts:
Checked - use interrupts
Unchecked - use polling mode

Packet Delay:
This parameter describes the Packet Delay in 16nSec Ticks between two IP-Data-Frames.
If this parameter is zero – XIS checks for an adequate Packet delay depending on the actual network speed and detector type using **Acquisition_GbIF_CheckNetworkSpeed** and passes a valid value to the library using **Acquisition_GbIF_SetPacketDelay**.
When the Parameter is above zero the value is set using **Acquisition_GbIF_SetPacketDelay**.
Valid numbers are e.g.
8000 - WLAN 420x3556 pixels, 5s transfer time preset value,
293 – LAN 420x3556 pixels, 500ms transfer time via 1GBit ~80% load,
1935 – LAN 1024 x 1024pixels at 15fps via a 1GBit connection
8000 – – LAN 1024 x 1024pixels at 1fps via a 100Mbit connection. ~80% load

Open via:
This parameter describes which parameter is passed to **Acquisition_GbIF_Init**. The detector can be opend by passing the MAC Address, the IP-Address or the Detector Name.

4.8.1.3 Configure GbIF

This dialog allows the user to change the IP Settings of the selected Detector if the detector and the Network adapter are in the same Subnet. In case of a XRpad the dialog will only show the current settings. Please use the integrated web-server to configure the ip-settings and the image-transfer-path (LAN/WLAN) of the XRpad.

Change IP-Settings:
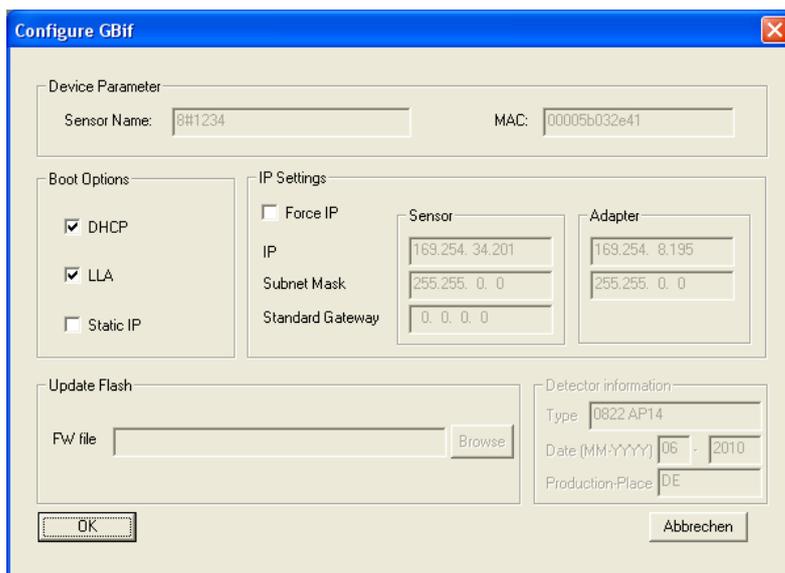
Figure 21: Configuration of the selected GigE-Detector

The user can set the mode to LLA (Local-Link Adress), DHCP (Dynamic Host Configuration Protocol) or Static IP – Address. LLA and DHCP can also be combined (see figure above). The changes will be valid after the Dialog is closed with "OK" and restart of the detector (power cycle).

Note: There has been discovered some issues. Please see **Trouble Shooting** for more information.

Force IP:

Figure 22: Configuration of the selected GigE-Detector – Force IP

If the detector has an address which cannot be opened due to incompatible network settings the user can force the device temporarily to connect with a custom IP-Address, but out of the same subnet and with the gateway like the network card of your computer system. For that check the Option "Force IP" (which can also be found within the "Config "dialog) and enter the temporary settings in the fields "IP", "Subnet Mask" and "Standard Gateway". After that press "OK".
After the addresses are set this way, PC and detector should be able to connect with each other. Now configure the boards permanent address settings as described above. With restart of the detector the device will loose the temporary IP and behave as configured (e.g. IP per DHCP or LLA or Static IP). **Please Note that this option is not available for the XRpad.**

4.8.1.4 Manual Setup

During the manual initialization the XIS asks for the communication channel to open, more than one are possible. The communication channel can be set by the hex-switch of the frame grabber board, if only one frame grabber inside the default channel is zero. The following dialog requests parameters to initialize the sensors connected to these channels.

Size of the detector

Define upper and bottom left and right X, Y dimension of the detector.

4.8.2 Sorting

The following sorting schemes are possible
(see also sorting overview):

> No Sort (e.g. XRpad, 0822, 1611, 1622 and 1642)
>
> XRD 0820/40
>
> XRD/RID 512-400
>
> XRD 1640 A / 1620 AJ
>
> XRD 1680 A
>
> XRD 162X AM/AN
>
> XRD 16X0 AN CS
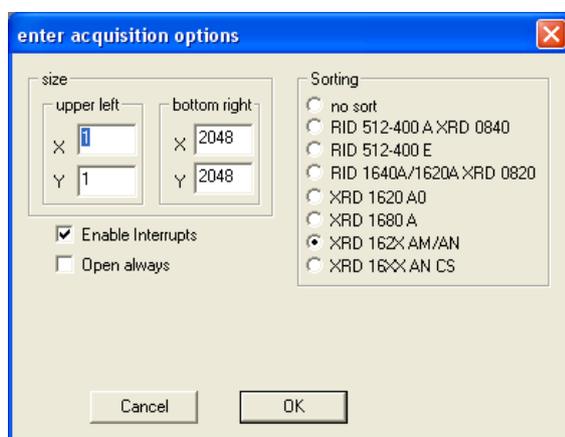
Figure 23: Dialog for the Acquisition Options

Enable Interrupts

The setting of different interrupt sources allow an efficient data transfer between I/O board and the memory of the PC (for further explanation see interrupt sources). If no interrupts are enabled the detector is running in polling mode. It is recommended to use the interrupted mode for an frame synchronization. (XRpad and xx22 detectors only provide the interrupt mode and will ignore this paramter)

Open always

This option opens the requested communication channel regardless if it has been already captured by another process running on the system. It isn't recommended to use this option except for debugging because the XISL can't free all resources in one process that were allocated by another process. If you initialized more than one sensor the XIS asks now a default detector to that all the further instructions correspond.

4.8.3 View

The View command comes to the "Enter View Options" dialog, where the presentation of the actual window can be optimized.

Toolbar: **x2**
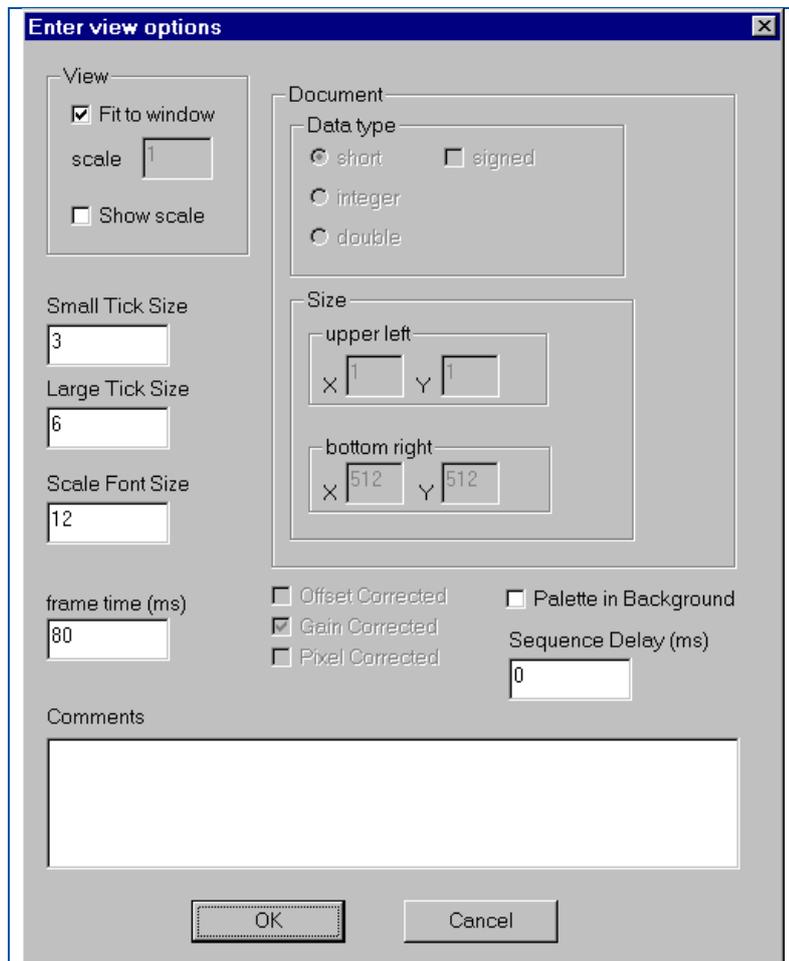
Figure 24: View Options Dialog

View – Fit to window	Defines the scale of the image window.
View – Show Scaling	To be switched ON for scaling the edges of the image
	Small Tick 2 (default)
	Large Tick 4 (default)
	Scale Font 12 (default)
Document – Data type	Information of the used data type.
	Short
	Integer
	Double
	Signed
Document – Size	Information on upper and bottom left and right X, Y dimension of the detector.

Table 22: Settings of the View Options Dialog

4.8.4 Options Active Sensor

The appearing dialog shows the connected sensors in a list box. The active sensor is highlighted. The sensors are identified by the frame grabber type they are plugged in and a unique number. To change the active sensor for acquisition another sensor can be selected from the list box.

Toolbar:

4.8.5 Options Sensor

This menu entry comes to the "Sensor Options" dialog contains information about the active sensor.

➤ The communication channel edit box gives information about the frame grabber board which is connected to the active sensor.

➤ In the correction directory the correction files are stored by the Save correction files command. If the Auto Correction Load radio button is checked suitable correction files are loaded from the correction directory at startup and if a different timing is selected.
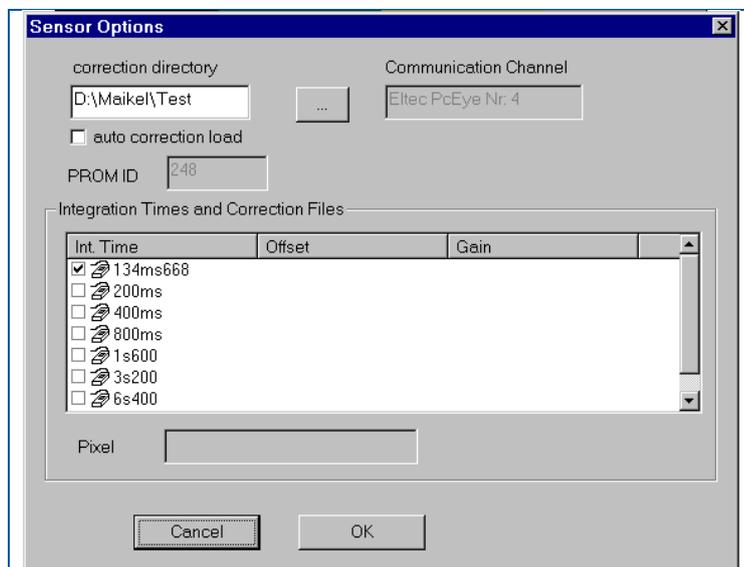
Figure 25: Sensor Options Dialog

➤ The PROM ID is the identifier of the actual used detector PROM. This information is important to get support at hardware problems concerning the detector or the frame grabber.

➤ In the group "Integration Times and Correction Files" all available integration times are listed. The selected one is checked in the list control. If correction files are attached for the different timings there will be corresponding entry in the list control. If a pixel correction file is attached it is displayed in the "Pixel" edit box.

4.8.6 Switch on/off Service Mode

Enabling the Service Mode means that special Service data is displayed into the first row of acquired images. These images can be analysed by PerkinElmer for the purpose of detector troubleshooting.

This function can also be implemented into customer application software (see 7.2.60 API function Acquisition_ ActivateServiceMode). The advantage for your application is that images can be stored in any raw data format (i.e. without compression!!) and, in spite of that, can be analysed by PerkinElmer. Currently this function is not supported by all detector types. (Refer to the API description for further details.)

4.8.7 Detector Options

The "Set Detector Options" dialog allows setting up the detector internal parameters:

➤ Binning-mode, Trigger-mode, Gain, Timing and Special TriggerMode are changeable as described in the Section 6-"Details for the Hardware".

➤ Use 'Data Delivered on Demand Options' to change the additional delay in DDD-Trigger Mode

➤ Use 'Select /TriggerOut Signal' to choose between different /TriggerOut signal options

➤ Use 'Select Region of Interest (ROI)' to select a specific region of the full field of view (Sectional Mode)
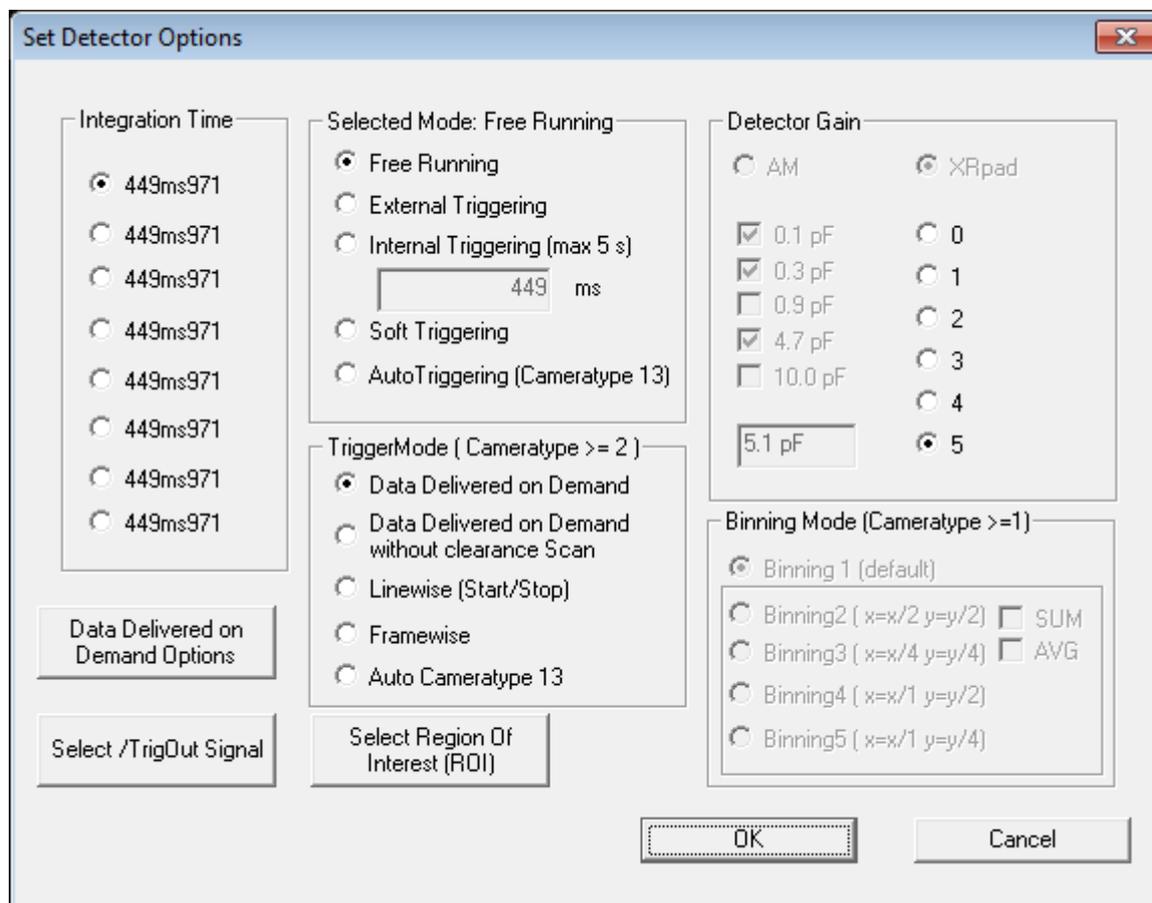
Figure 26: Set Detector Options

4.9 Help

4.9.1 Contents

This command displays the About Box.

4.9.2 About

This command displays the version number and copyright notice of XIS.

4.10 Toolbar

The toolbar is displayed across the top of the application window and below the menu bar. The toolbar provides quick mouse access to many tools used in XIS.

The Toolbar can be hide or displayed by the command Toolbar in the View menu (ALT+ V, T).

Click	Application
	Open a new document.
	Open an existing document. XIS displays the Open dialog to find and select the files.
	Save the active document or template with its current name. If the file is not named XIS displays the Save As dialog box.
	Print the active document.
	Copy the selection to the clipboard.
	Insert the contents of the clipboard at the insertion point.
	Display the About Box.

Table 23: Windows Toolbar

4.11 Acquisition Toolbar

The toolbar is displayed across the top of the application window and below the menu bar. The toolbar provides quick mouse access to many tools used in XIS.

The Toolbar can be hide or displayed by the command Toolbar in the View menu (ALT+ V, A).

Click	Application
	Starts the continuous Acquisition. **Shortcut:** CTR+SHIFT+C
	Opens the Sequence Dialog
	Starts a Single Shot
	Opens the Player Dialog.
	Opens the View Dialog
	Opens the Active Sensor Dialog

Table 24: Acquisition Toolbar

4.12 Status Bar

The status bar is displayed at the bottom of the XIS window. To display or hide the status bar the Status Bar command in the View menu can be used.

The left area of the status bar shows online information for the menu or Tool bar entry where the cursor is above or shows the status of an executed command.

The right areas of the status bar give extended information regarding acquisition status, pixel values and frame times.

Indicator	Description
Box 1:	Describes the main status of the software. (Online Help)
Box 2:	Warnings by using detector and its messages.
Box 3:	Selected X-value of the detector array.
Box 4:	Selected Y-value of the detector array.
Box 5:	Actual value in digits between 0 – 65535.
Box 6:	Marker for Offset ON/OFF.
Box 7:	Marker for Gain/Offset ON/OFF.
Box 7a:	Marker for GainSeq/Offset ON/OFF.
Box 8:	Marker for Pixel correction ON/OFF.
Box 9:	Selected frame time.

Table 25: Status Bar

4.13 Title Bar

HIS - [test.his]   _ □ ✕

The title bar is located along the top of a window. It contains the name of the application and document. To move the window, drag the title bar Note: You can also move dialog boxes by dragging their title bars.

A title bar contains the following elements:

➤ Application Control-menu button
➤ Document Control-menu button
➤ Maximize button
➤ Minimize button
➤ Name of the application
➤ Name of the document
➤ Restore button

4.13.1   Scroll bars

Displayed at the right and bottom edges of the document window. The scroll boxes inside the scroll bars indicate your vertical and horizontal location in the document. You can use the mouse to scroll to other parts of the document.

4.13.2   Size command (System menu)

This command displays a four-headed arrow to size the active window with the arrow keys. This command is unavailable if the window is maximized.

After the pointer changes to the four-headed arrow:

1. Press one of the DIRECTION keys (left, right, up, or down arrow key) to move the pointer to the border you want to move.
2. Press a DIRECTION key to move the border.
3. Press ENTER when the window is the size you want.

Shortcut

Mouse: Drag the size bars at the corners or edges of the window.

4.13.3   Move command (Control menu)

This command displays a four-headed arrow to move the active window or dialog box with the arrow keys. This command is unavailable if the window is maximized.

Shortcut

Keys: CTRL+F7

4.13.4   Minimize command (application Control menu)

Use this command to reduce the XIS window to an icon.

Shortcut

Mouse: Click the minimize icon [▼] on the title bar.

Keys: ALT+F9

4.13.5   Maximize command (System menu)

Use this command to enlarge the active window to fill the available space.

Shortcut

Mouse: Click the maximize icon [▲] on the title bar; or double-click the title bar.

Keys: CTRL+F10 enlarges a document window.

4.13.6   Next Window command (document Control menu)

Use this command to switch to the next open document window. XIS determines which window is next according to the order in which you opened the windows.

Shortcut

Keys: CTRL+F6

4.13.7   Previous Window command (document Control menu)

Use this command to switch to the previous open document window. XIS determines which window is previous according to the order in which you opened the windows.

Shortcut

Keys: SHIFT+CTRL+F6

4.13.8   Close command (Control menus)

Use this command to close the active window or dialog box.

Double-clicking a Control-menu box is the same as choosing the Close command.

Note: If multiple windows opened for a single document, the Close command on the document Control menu closes only one window at a time. All windows can be closed at once with the Close command on the File menu.

Shortcuts

Keys: CTRL+F4 closes a document window

        ALT+F4 closes the current window or dialog box

4.13.9   Restore command (Control menu)

Use this command to return the active window to its size and position before you chose the Maximize or Minimize command.

4.13.10   Switch to command (application Control menu)

Use this command to display a list of all open applications. Use this "Task List" to switch to or close an application on the list.

Keys: CTRL+ESC

Dialog Box Options of the Switch command

Task List	The next application can be selected.
Switch To	Makes the selected application active.
End Task	Closes the selected application.
Cancel	Closes the Task List box.
Cascade	Arranges open applications as overlapped windows whereby the title bars are visible. This option has not effect on applications reduced to icons.
Tile	Arranges open applications into not overlapping windows. This option has no effect on applications reduced to icons.
Arrange Icons	Arranges the icons of all minimized applications across the bottom of the screen.

Table 26: Switch Command Dialog

4.14 Standard Dialogs

4.14.1   File Selection Dialog

This dialog shows all available data files stored on hard disk or loaded in the RAM. This dialog appears by the commands Link Offset Correction, Link Gain Correction and Link Pixel Correction.
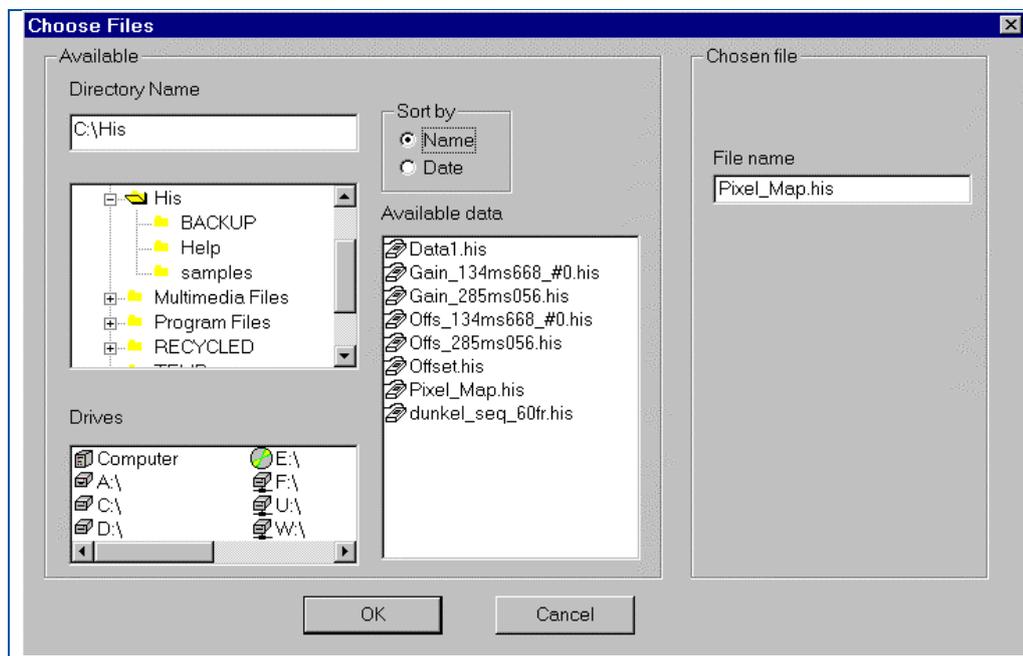
Figure 27: Choose File Dialog

The available group box shows all available drives, directories and files. The currently loaded data's are listed below the drive "computer".

By pressing one of the "sort" radio buttons one can influence the way the available data are sorted in the "available data" list box.

Double click the selected file from the "available data" box and the file will be displayed in the "Chosen file" box. The process will be executed by pressing the "Ok" button or the "enter" key

4.14.2 Overwrite data dialog

This dialog appears if new data has to be acquired into an unsaved window.
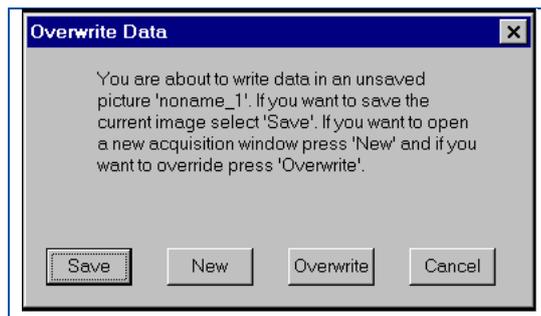
Figure 28: Overwrite Data Dialog

Save	Saves the current data and writes the new data in the window buffer
New	A dialog appears to define the new document type
Overwrite	Overwrites the existing data without any saving
Cancel	Cancels the dialog

4.14.3 Choose Directory Dialog

This dialog appears if a directory has to be selected.
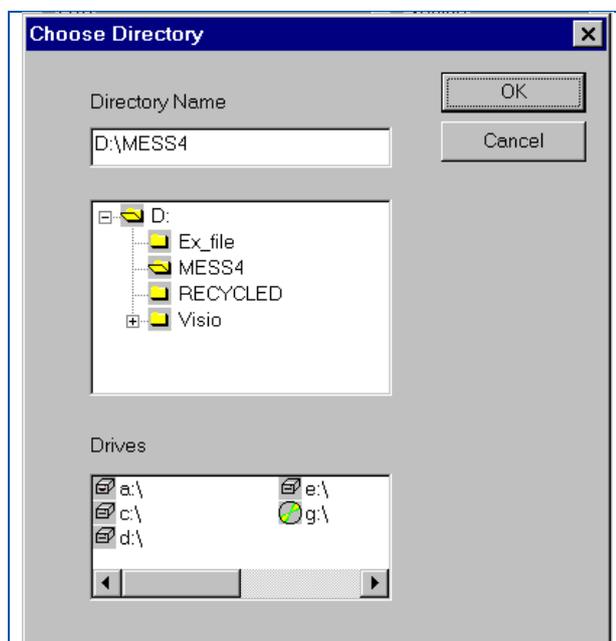
Figure 29: Choose Directory Dialog

directory name	This edit line specifies the selected directory. It's connected to a directory tree and all current available folders are listed.
drives	All current available drives are listed here. If a drive is selected the directory name and its directory tree is actualized.

5 Application

5.1 Mathematical Expressions

This dialog enables the easy input of mathematical expressions for image processing.
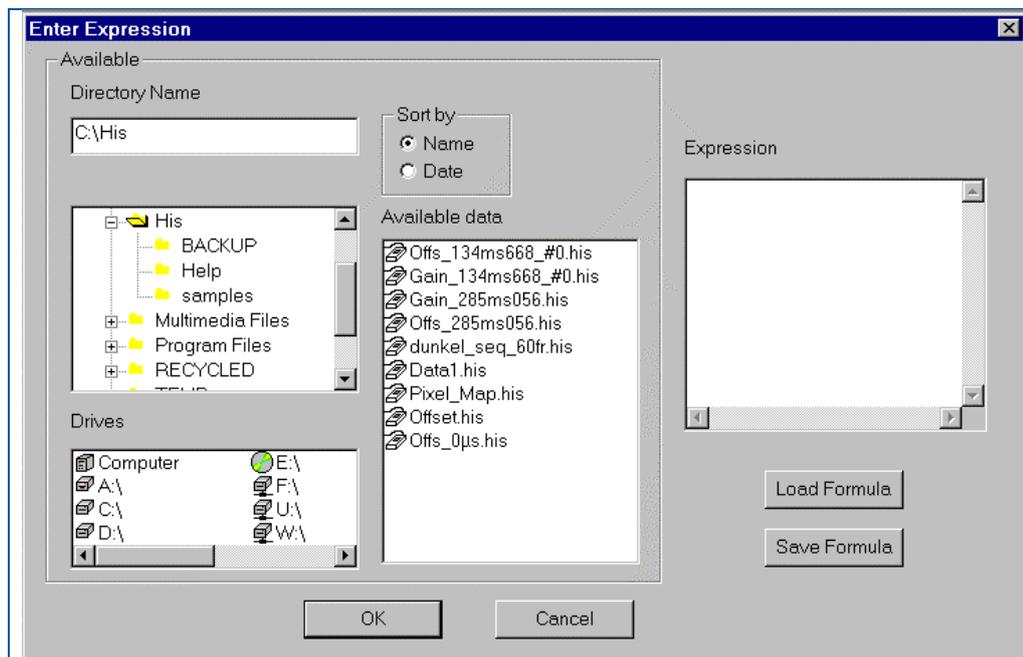
Figure 30: Mathematical Expressions Dialog

The available group box shows all available drives, directories and files. The currently loaded data's are listed below the drive "computer". To list stored images click on the items of the drive box and the entries of the above tree control displaying the directory structure.
By pressing one of the "sort" radio buttons one can influence the way the available data are sorted in the "available data" list box.
The mathematical expression can be added in the "Expression" edit box. Images can simply inserted by drag and drop files from the "available data" list to the edit box. The file name is inserted at the current cursor position. Soft line breaks can be set by pressing the "enter" key while holding the "Control" key.

5.1.1 Parsing expression:

Reserved symbols are:

+	Addition operator
-	Subtraction operator
*	Multiplication operator
/	Division operator
(	open parentheses
)	close parentheses
=	Assign operator
...	dots used to specify a range of frames in a data sequence
[	open square brackets to specify a range of frames in a data sequence
]	close square brackets to specify a range of frames in a data sequence
(USHORT)	cast operator, converts an arbitrary data type to the required data type (unsigned short)
SUM	summation function
AVG	Average function.
ABS	Absolute function

Table 27: Parsing symbols

+ operator:

If both operands are numbers the result is also a number. If one operand is a number and the other is an image then the number is added to every data entry. If both operands are images the data are added pixel by pixel.

- operator:

If both operands are numbers the result is also a number. If one operand is a number and the other is an image then the number is subtracted from every data entry. If both operands are images the data are subtracted pixel by pixel.

* operator:

If both operands are numbers the result is also a number. If one operand is a number and the other is an image then the number is multiplied with every data entry. If both operands are images the data are multiplied pixel by pixel. A matrix multiplication is not performed!!!

/ operator

If both operands are numbers the result is also a number. If one operand is a number and the other is an image then the number is divided by every data entry. If both operands are images the data are divided pixel by pixel. If a division by zero is recognized the parser returns with an error message.

= operator

If the result of an arithmetic expression is an image then this result has to be assigned to a result window.

For instance: Dest.his = data1+40000

is a valid expression. To the data entries of "data1" a value of "40000" is added and the result is written into "Dest.his".

The expression *Dest.his = 40000*

returns an error because the result is a number.

The expression data1+50000-data2

also causes a parse error because the result is an image and the assignment to a result window is absent.

(Type) (cast operator)

This operator converts an arbitrary data format into the required ones. Type can be one of the following words:

SHORT	signed 16 bit integer
USHORT	unsigned 16 bit integer
LONG	signed 32 bit integer
ULONG	unsigned 32 bit integer
DOUBLE	8 byte floating point number

Table 28: Type Operators

Dest.his = (ULONG) (data1+data2/16-SUM(FRAMES, data5[2...4]))

[a...b] operator (range operator)

Syntax: Data.his[a...b]

This operator returns a sequence of b-a frames extracted from the source sequence (here Data.his) starting at frame a and ending at frame b.

Dest.his = data1[3...7]

SUM function:

The sum function derives the sum of numbers, the sum of different images, the sum of rows or the sum of columns of different images. The entries in this function are separated by comma. The first parameter has to be one of keywords.

FRAMES	Derives the sum of different frames.
ROWS	Derives the sum of different rows. If the following arguments represent more then one frame, the result is a sequence of frames containing the summed rows of the frames.
COLUMNS	Derives the sum of different columns. If the following arguments represent more then one frame, the result is a sequence of frames containing the summed columns of frames.

Table 29: SUM - function parameters

If an entry is a sequence of several frames the sum of the sequence is evaluated and after that the resulting images are summed. Valid expressions are for instance:

Dest.his = SUM(FRAMES, data1[3...6], data2, 40000, -30000)

Dest.his = SUM(ROWS, data1[3...6], data2, 40000, -30000)

Dest.his = SUM(COLUMNS, data1[3...6], data2, 40000, -30000)

AVG function:

The average function derives the average of numbers as well as the average of different images or both. The entries in this function are separated by comma. If an entry is a sequence of several frames the average of the sequence is evaluated and after that the resulting images are averaged. Valid expressions are for instance:

Dest.his = AVG(data1, data2, 40000, -30000)

ABS function:

The absolute function derives the absolute values of numbers or data entries. A valid expression is for instance:

Dest.his = ABS(data1)

General remarks:

All data or numbers are converted to floating point numbers before the expression is evaluated.

The result images contain floating point data (if not casted).

5.2   Image Correction

The X-Ray Detectors (**XRD**) needs an Offset correction to take into account the dark current of each pixel. In particular, during the warm-up phase of the detector, the Offset is not stable and during this time period the detector use is not recommended. During operation it is recommended to refresh periodically the Offset.

Additionally, a Gain correction is necessary to homogenize differences in pixel sensitivities or to take into account the X-ray beam illumination; therefore it is very important that the whole image area is illuminated homogenously. The Gain correction should be carried out in an optimum dynamic range of the sensor (70-80 % of the full scale range FSR) or in the dynamic range of interest. The radiation intensity used to create the gain image can depend on the application, e.g. if the typical grey level is about 10.000 ADU and the remaining area is saturated, it is recommended to use a gain image created at 10.000 ADU. The use of an Offset and Gain calibration eliminates offset dependency and therefore any stored Gain correction file can be used for a specific frame time for longer time periods.

The image performance can be enhanced by using the Multiple Gain Correction. For each dynamic range of interest a separate offset corrected and averaged gain image is used as an interpolation point. The maximum number of interpolation points depends on the installed computer memory and the **XRD FG[X][e] Opto** frame grabber can operate with up to 10 gain points (offset corrected bright images). It is important that each gain image is completely and homogenously illuminated. The radiation intensity used to create the gain images can depend on the application, e.g. if the typical grey level is between 5.000 ADU and 10.000 ADU and the remaining area is saturated, it is recommended to use gain images created at 5.000 ADU, 10.000 ADU and 45.000 ADU.

The Pixel correction allows a 'software repair' of underperforming pixels to enhance image quality. Underperforming pixel values are replaced with the averaged value of the surrounding eight adjacent pixels where underperforming pixels are not used. The pixel correction is only performed on specific pixels, mapped in the file PXLMASK.HIS. The delivery package includes the PXLMASK.HIS file for the specific detector; the user can also generate their own correction file. Please be aware that the number of pixels used for the mean correction should be minimized. The pixel correction procedure requires CPU time (**XRD FG[e]** Frame Grabber / **GbIF**) and depending on its speed, a slower presentation of the acquired images on screen might occur in relation to the selected timing mode of the system. The main screen of the **XIS** software sends out a warning message displayed on the screen if all of the acquired images are not accepted by the computer.

All Corrections are very similar for the **XRD FG[e]/GbIF** and **XRD FG[X][e] Opto** Frame Grabbers. All procedures can be used in the same manner and the XRD driver will automatically select the software or hardware correction.

5.2.1   Use of the Offset Correction

The offset correction of images is recommended to eliminate the influence of pixel dark currents in the acquired image. To get an Offset correction file the following steps have to be performed:

1. Select the desired integration time, readout mode and gain setting.
2. Switch off the X-ray source so that the detector only transfers its "dark image".
3. Wait a few seconds until the detector achieves an equilibrium.
4. Start the Get Offset Image. / Start All Offset Images.
5. Select a number of frames.
   It is recommended to use between 20 to 100 frame cycles which will be averaged. The averaged image is qualified as the new Offset Image of the selected frame time and automatically linked to later acquired images.
6. Control the new acquired image using the Options/View command and/or Brightness, Contrast or LUT range.
7. The Offset correction file can be saved if desired.
   Note: A warning appears if the program is left without saving new acquired Offset correction files.

The Offset correction should be repeated periodically. In particular during the warming-up period of the system, the dark current of the pixels may change considerably.

To interrupt the procedure the <ESC> key can be used.

5.2.2   Use of the Gain/Offset Correction

The Gain/Offset correction of images is recommended to eliminate the influence of pixel sensitivities and influences of the used X-ray source in the acquired image. To get a Gain/Offset correction file the following steps have to be performed:

1. Select the desired integration time, readout mode and gain setting.

2. Acquire a new Offset correction image.

3. Switch on the X-Ray source and control the brightness of the acquired image in the desired read out settings. The detector's acquired intensity should be between 70-80 % of FSR or in the range of the ROI. The gain intensity depends on the application, but the whole image area should be illuminated homogenously.

4. Start Get Gain/Offset Image.

5. Select a number of frames.
    It is recommended to use between 20 to 100 frame cycles which will be averaged. The averaged image is qualified as the new Gain/Offset Image of the selected frame time and automatically linked to later acquired images.

6. Control the new acquired image by using the Options/View command and/or Brightness, Contrast or LUT range.

7. Store the Offset correction file if desired.
    Note: A warning appears if the program is left without saving new acquired Offset correction files.

To interrupt the procedure the <ESC> key can be used.

NOTE: The Gain image is automatically Offset corrected with the currently linked Offset correction file. To get best quality of the correction file, please perform a new Offset correction before starting Gain/Offset correction. If the number of averaged frames for the gain file is too small then the limited SNR is also limiting the SNR of the corrected image.

5.2.3   Use of the Multiple Gain Correction

The Multiple Gain Correction is recommended to eliminate the influence of pixel sensitivities and influences of the used X-ray source in the acquired image. To get a Multiple Gain Correction file the following steps have to be performed:

1. Select the desired integration time, readout mode and gain setting

2. Acquire a new Offset correction image.

3. Switch on the X-Ray source and control the brightness of the acquired image in the desired read out settings. The detector's acquired intensity should be in the range of the ROI. the whole image should be illuminated homogenously.

4. Start the Acquire Sequence.

5. Select a number of frames and the average mode.
    It is recommended to use between 20 to 100 frame cycles which will be averaged.

6.   Control the new acquired image by using the Options/View command and/or Brightness, Contrast or LUT range.

7. Store the Offset corrected bright image.

8. Repeat the steps 3-7 for each intensity of interest.

9. Create a Gain–Sequence with Acquire/Build Gain Sequence

10. Start a new acquisition.

11. Link the created Gain-Sequence file with Acquire/Link Gain Sequence

12. Store the Offset correction file if desired.

Note: A warning appears if the program is left without saving new acquired Offset correction files. If the number of averaged frames for the gain files is too small then the limited SNR is also limiting the SNR of the corrected image.

5.2.4   Use and generation of the Pixel Correction

The Pixel Correction of images is recommended to eliminate the influence of underperforming pixels of the detector in acquired images. To get a Pixel correction file the following steps have to be performed:

1. Select the desired integration time, readout mode and gain setting.
2. Link correction files (Offset, Gain).
3. Switch on the X-ray source and in the continuous mode, control the brightness of the acquired image. The detector's acquired intensity should be between 70-80 % of its maximum signal.
4. Start an image acquisition as for the Get Gain/Offset Image (no sample in front of the detector).
5. Control the new acquired image by using the Options/View command and/or Brightness, Contrast or LUT range.
6. The window should show a homogenous corrected image. Intensity deviations are a sign of not fully working pixels.
7. Change the x-ray source such that the intensity acquired by the detector should be about 50 % of its maximum signal.
8. Go to Select by Value in the Edit Menu.
9. Enter desired range of good pixels (e.g. 15000-45000 out of 0-65535).
10. Select "Out of range" button. (All selected pixels are marked.)
    You can also select pixels using "strg+ LeftMouseButton".
11. Call Create Pixel Map in the Edit Menu.
12. The Pixel Map is created and can be stored as new PXLMASK.HIS.

NOTE: The new PXLMASK.HIS is automatically linked to new acquisitions and the acquired start-up image (see 4.) is also corrected. Beware that each binning mode requires an individual PixelMask.

5.2.5   Correct already acquired images

It is possible to correct already acquired uncorrected images. Select the desired image by the Window Command and use one of the Link Commands (Link Offset Correction, Link Gain Correction, Link Gain Sequence Correction or Link Pixel Correction). The active image is automatically corrected.

These setting are also used for the next acquisitions.

NOTE: It is not recommended to close linked correction files during a running acquisition. This can lead in a closing application.

5.3   Acquisition Control Modes

Five different acquisition control modes are available. The **Free Running** mode is the default mode which means that the detector sends out continuously frames according to the selected frame time. The **External Trigger** mode means that the detector sends out a frame after triggering by an external pulse and ignores all other incoming trigger pulse until the selected frame time has elapsed. After that the detector can be triggered by a new pulse. Details of the trigger pulse are described in the chapter External Trigger. The third mode is the **Internal Trigger** mode. In this case a fixed pulse frequency between the fasted free running mode of the detector and 5 seconds in steps of 1μs can be selected. These pulses are send via frame grabber board to the detector or , in case a GigE detector is used, generated in the detector using the incternal clocks.

The fourth mode is the **Soft Trigger** mode. In this case one single pulse is send from the workstation to the detector generated by software via Network/Optical/HIIB I/F. The fifth mode is the **Auto Trigger** mode (only available with theXRpad detectors series). In this mode the detector acquires an image when a defined exposure level is reached.

The control mode can be selected in the submenu **Mode** and the integration time can be selected in the submenu **Timings** of the menu **Detector**. The selected integration time and mode are marked by a check mark on the right side of these items. The trigger pulse can be send as well to the PC frame grabber boards **XRD FG[e]** and **XRD FG[X][e] Opto** as to the detector (see: External Trigger) or the Interface Power Unit (XRpad IPU).

The trigger modes are recommended if a pulsed x-ray source is used. If the x-ray pulse appears during the readout time of the detector the information are split into two frames. Also if these

frames are summarized there could be artefacts which are not correctable. The trigger mode realizes an expose during the delay of the readout structure.

To start external triggering the following steps have to be performed:

1. Connect the trigger cable with the detector or with the frame grabber board
2. Power on the detector
3. Startup **XIS**
4. Select an initialization of the detector (Interrupt or Polling Mode)
5. Select the desired integration time (Timing Menu)
6. Select the **External Trigger** Mode
7. Link adequate correction image (Offset, Gain/Offset and PixelMask) for the current detector settings (Timing/Gain/Binning).
8. Send trigger pulses
9. Start the desired acquisition mode (Continuous or Sequence)
10. Send trigger pulses

In case of internal triggering the steps are similar to the free running mode. To start the internal trigger mode the mode has to be selected and the frequency has to be inserted in the appearing dialog. After that the detector sends out frames in the desired frequency and the frames can be acquired continuously, in a sequence or as a single shot.

Note: In the external trigger mode the detector is waiting for a new acquisition until the trigger pulse is sent. During the desired frame time a new trigger pulse has no effect. The correction files have to be created with the same frequency of trigger pulses for best results.

Example: If the detector is set to the integration time 400ms and the trigger pulse is sending every 200ms, the detector nevertheless runs with >=400ms. But if the pulse appears every 450ms the detector runs with 450ms. The selected integration time should always below the desired period time of the trigger pulse frequency. The lower limit is the shortest free running timing (Timing 0) which can be selected.

5.4 Warning table by using the detector and its status

Warning: You are loosing frames
Based on CPU speed and used correction mode, the monitor can not present all received images from the detector.
=>   Change to a longer integration time per frame or use less on-line corrections.

Black or white value out of range
If the user selects values out of the allowed range of 0 - 65535 digits. This is not allowed.

Board initialization failed
Starting the software, the detector could not be initialized.
=> Check power cords and interface cables and restart Acquisition in the Options menu.

Board initialization successful
The I/O board was successfully initialized, continue with Acquiring images.

Not all functions available
This message appears if no detector or no I/O board can be detected. The XIS software can be used for image presentation of stored images and further processing of these images.
In the manual initialization of the detector some additional features are also not available.

Acquisition done
This message appears if a started acquisition of images is done. The user can continue using further XIS commands.

Ready
This message appears if one of XIS commands is done. The user can continue using further XIS commands.

6   Details for the Hardware

6.1   Readout schema

6.1.1   Free Running

Generally the detector is in free running mode after powering up. The detector will automatically perform in its fastest full resolution in timing T0 and continuously scan the images. The other available timings T1 up to T7 differ from T0 in an increased time in between frames to artificially reduce the frame rate and to extend the integration time.

For example the first timing (Timing 0) of the XRD0820 detector needs a minimum of 132.977 ms for one frame. This means that each pixel is read out every 132.977 ms. During this time the pixel collects radiation. The longer timings (Timing 1-7) consist of a readout followed by a delay. The delay time is the time of the selected integration time minus the time of the first timing (Timing 0). As an example the timing two of the XRD0820 is 199.708 ms, this means the first 132.977 ms stand for the readout of the detector. The following 66.731 ms is an additional delay in which the detector is only integrating radiation.

Some detector versions have different row types for the timings (1-7) to enhance the image quality at lower speed. For more details concerning the different readout timings see the detector manual. The integration time of one frame for each timing can read out with the function Acquisition_GetIntTimes(..) or it can be calculated as described in chapter 7.2.38 CHwHeaderInfo. To realize a delay using the fastest readout the internal trigger mode can be used with a frame time of the selected timing plus delay time. For more details see the paragraph Internal Trigger.

Note: If a pulsed x-ray source is used it is recommended to expose during the delay of the detector. If the x-ray pulse appears during the readout time the information is split into two frames. Also if these frames are summarized there could be artifacts which are not correctable. To realize an exposure during the delay the detector allows the triggering of the x-ray source and the detector itself.

6.1.2   External Trigger

Triggering the detector is the attempt to synchronise the detector to other devices e.g. x-ray sources having their specific schemes of radiating x-ray pulses. The current mode is triggering the detector on a frame by frame basis which means that the detector sends a frame after triggering by an external pulse and ignores all other incoming pulses until the selected frame time has elapsed. After that the detector can be triggered by a new pulse.

In order to trigger the detector a 20µs wide low active trigger pulse has to be transmitted to the device. The trigger signal has to be generated externally and can then be connected to either pin one of the 7-pin round connector (/TRIG_IN) located directly at the detector device or connected to the sub-click located on the rear side of the **XRD-FG** (/TRIG_IN converted to RS422 signal as /FR_SYNC) respectively the D-Sub connector of the **XRD-FG[X][e] [Opto]** (/TRIG_IN as LVDS signal converted to /FR_SYNC). (For connector details refer to 6.1.9)

Trigger pulses are accepted from both sources. Prior to this the detector has to be set per command into the external trigger mode. The waveform of the trigger pulse as shown in chapter 'Trigger-modes' describes the triggering mode on a frame by frame base. The period of the trigger waveform determines the integration time.

6.1.3   Internal Trigger

The internal trigger mode works similar to the external trigger mode and is also triggering the detector on a frame by frame basis which means that the detector sends a frame after it has been triggered by a trigger pulse and ignores all other incoming pulses until the selected frame time has elapsed. After that the detector can be triggered by a new pulse.

The trigger pulse is either delivered by the frame grabber or generated by an internal clock (GigE Detectors). It is a fixed integration time between the fastest free running mode of the detector and 5 seconds in steps of 1µs. The frame grabber sends the signal /FR_SYNC over the XRD Interface Bus or the Optical Interface to the detector.

Note: Due to internal processing the start of frame a frame after the trigger pulse was recognized can jitter ~2µs.

6.1.4 How to use the internal trigger mode

The following method describes how the internal trigger can be used to implement different integration times or to use one of the readout schemes with a customized delay. If the interface is connected and the power is switched on, the detector runs in the first free running mode. To use the internal trigger mode the following steps have to be performed:

1. The detector has to be set to the desired readout time using the Timings menu
   XISL: Acquisition_SetCameraMode(*hAcqDesc*, 0..7)
   -> The detector runs continuously (free running) in the desired timing and readout scheme.

2. The detector has to be set to the internal trigger mode using the Detector Mode menu
   XISL: Acquisition_SetFrameSyncMode(*hAcqDesc*, HIS_SYNCMODE_INTERNAL_TIMER).
   -> The detector aborts the current frame and waits for a trigger signal.
   -> The shortest repeat time of an trigger pulse is the selected timing (readout time).

3. Select the frame rate (readout plus delay time) in the appearing dialog (readout time … 5s).
   XISL: Acquisition_SetTimerSync(*hAcqDesc*, integration_time).
   The frame grabber sends the /FR_SYNC signal directly via the HIIB/Optical Link to the detector in the selected integration time or generates an internal signal in the selected integration time (GigE Detectors).
   -> The detector runs "continuously" with the selected integration time.

4. Start the acquisition
   XISL: Acquisition_AquireImage(*hAcqDesc*, *frames*, …)
   -> The frame grabber grabs the data into the memory.

6.1.5 Software Trigger

The software trigger mode works similar to the external trigger mode. It sends a Trigger Signal to the detector which will start a readout on base of the selected trigger mode (Framewise, DDD, Linewise, Auto)

6.1.6 Auto Trigger[4]

The Auto-Trigger is a unique acquisition mode as it detects the presence of applied x-rays by itself. Upon detection of an x-ray exposure the firmware halts the XRpad for a predefined time to allow for a longer x-ray exposure interval and then resumes operation deriving the image. After acquiring an image the detector returns into the condition where it checks for the next x-ray exposure. Thus it is possible to acquire multiple images.

It is also permissible to use the Auto-Trigger as acquisition mode and to select any other trigger source i.e. the soft trigger to trigger the firmware in absence of real x-rays in order to artificially retrieve an image for offset correction.

In order to invoke the Auto-Trigger the Auto-Trigger needs to be selected as acquisition mode and secondly the signal that is generated when x-rays are detected needs to be selected as trigger source, similarly as selecting any other trigger source to artificially trigger the firmware.

---

[4] Only for XRpad detector series

6.1.7   Trigger-modes

6.1.7.1   Frame-wise (default)

This is the most general trigger mode. Once the detector has been set to trigger mode the detector will synchronize to trigger events and perform a complete detector scan and transmit one image. This is a frame wise trigger mode which allows the operator to take control of the integration time of the detector. Trigger events during the scan process are ignored. Thus the shortest possible integration is ideally equivalent to the integration time of the fastest mode in free running.

6.1.7.2   Data Delivered on Demand triggered mode

Trigger with DDD mode is available for detectors with HeaderID 14 and CameraType >2
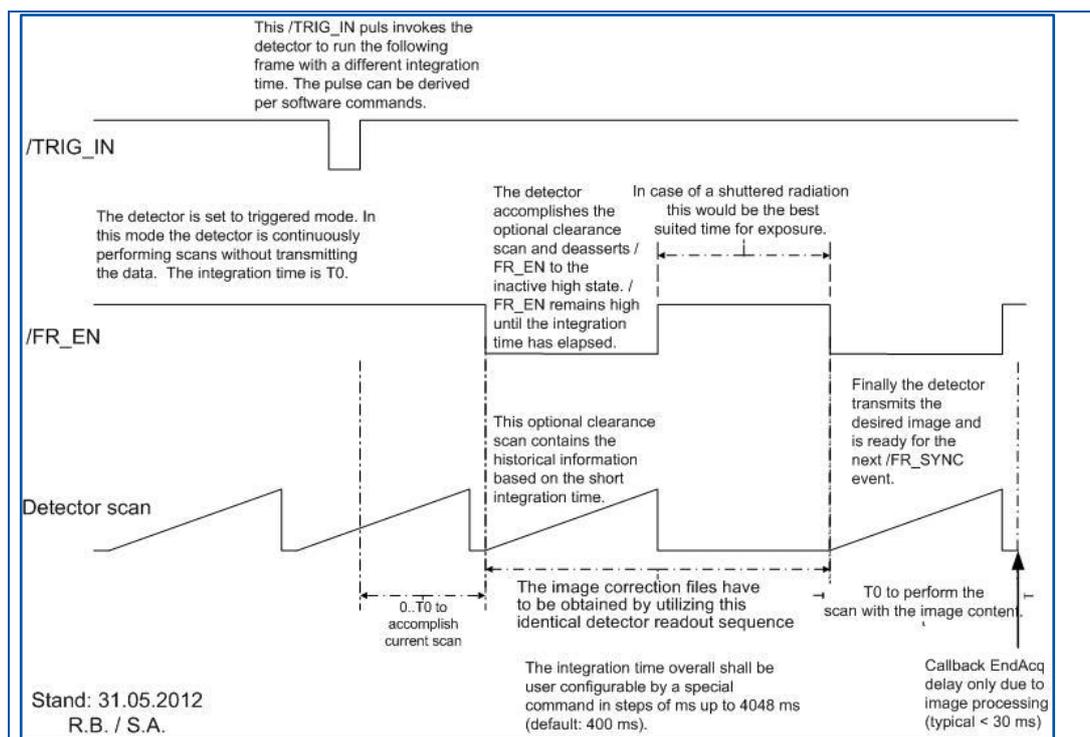
Figure 31: timing diagram for 'Data Delivered on Demand' triggered mode

The detector is running in a "silent" mode, like free running mode but without transferring image data. If the user application (Software, Frame Grabber or external Source) sends a triggers signal to the detector, the detector accomplishes the current frame. The next frame (clearance scan) is processed also with the fastest integration time. After that the detector waits until a customer defined time has elapsed (delay time). In this time gap the detector shall be exposed in case of pulsed or shuttered radiation. After the delay time the detector performs the image scan and transfers the desired data. After this the detector returns to the silent mode until the next trigger pulse is sent.

Note: If desired, the transmission of the clearance scan can be skipped. For that chose Trigger mode 1 with Acquisition_SetCameraTriggerMode(..) (refer to chapter 7.2.55). In this mode xx22 / 1642 Detectors skip the read out and the transmission of the clearance scan.

6.1.7.3   Start/Stop trigger-mode

One approach to use the detector in conjunction with pulsed x-ray sources is the start/stop trigger mode. It provides the possibility to improve the detector performance by synchronizing the detector operation to the pulses on a line or a group of line scans per trigger event basis. It is the attempt to fit a suitable amount of line scans in between consecutive pulses. The goal is to achieve good detector performance by doing offset- and gain corrections in a highly synchronized manner on a line by line basis

At first the detector is on halt and is waiting for the /TRIG_IN signal to transition to the active low state. Once this has happened there will be a delay based on internal processing of preceding lines until eventually the first real image line scan occurs. Due to the pipeline architecture of the detector it takes another cycle until the first digital converted data can be transmitted to the computer host system.
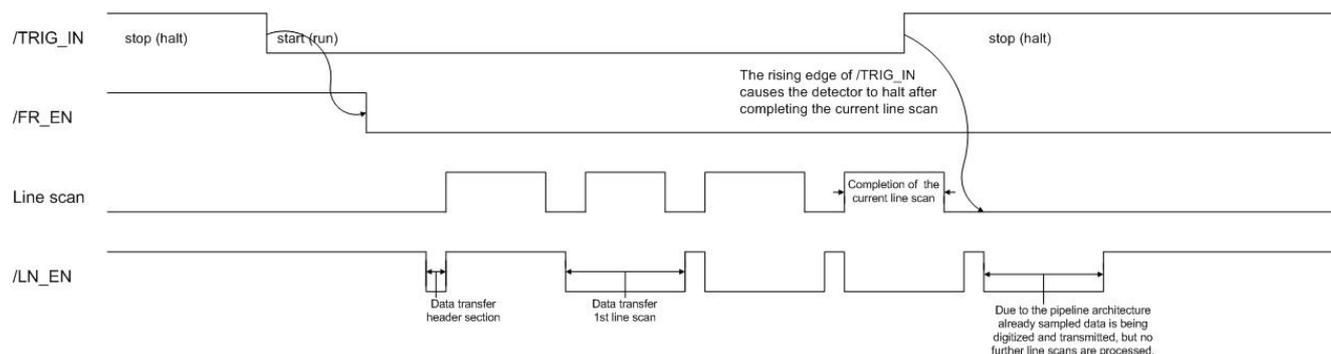
The detector will continue consecutive line scans until the /TRIG_IN signal is released again. Upon this the detector will complete the current scan activity and change to halt mode. The already sampled data will nevertheless be digitized and transmitted. The detector will reside in halt mode until the next transition of the /TRIG_IN signal to the low state which is described in the following picture:
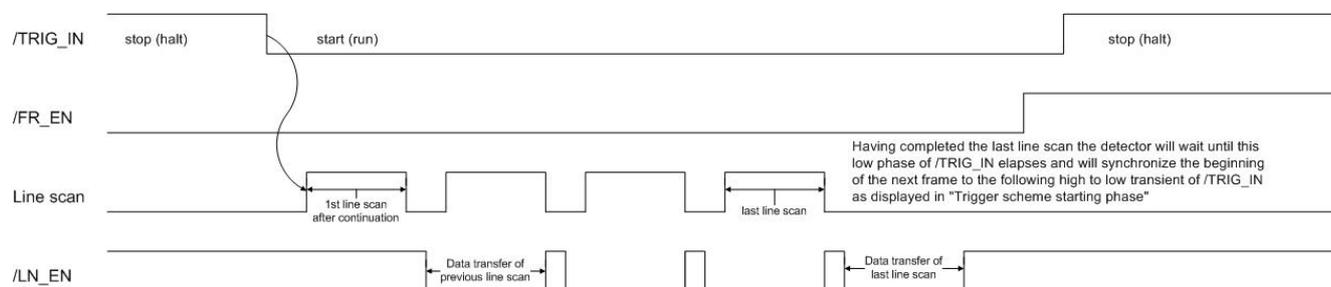
Towards the end of the image acquisition when having the last scan line completed the detector will wait until /TRIG_IN gets released and will resynchronise the beginning of the next frame to a new trigger sequence. This prevents static image distortions to scroll through sequences of images giving a chance to offset and gain correct.

It is the user's task to provide the run / stop signal as /TRIG_IN taking into consideration that the system has a latency of up to a row (line) read time to go into halt. In order to achieve less image distortion it is important to keep the stop phase as short as possible.

6.1.7.4   Auto Trigger
Please refer to 6.1.6

6.1.8   Trig Out Options

The trigger output signal, of the detectors trigger connector can be driven by multiple sources. Please refer to the detector manual whether the detector type you have attached can switch the TriggerOut Source.

These options can be set using the library function Acquisition_SetTriggerOutSignalOptions(..) All XRD 0820 AN, XRD 1620 AN, XRD 1621 AN, XRD xx22 AO/AP and 1642 detectors support the function.

This is the complete listing of available Trigger Output Signal assignments:

0:	FRM_EN_PWM (default)
1:	FRM_EN_PWM_INV
2:	EP (Exposure Pulse)
3:	EP_INV
4:	DDD_PULSE
5:	DDD_PULSE_INV
6:	GND
7:	VCC

Independent from the operating mode the trigger output selections 0, 1, 2, 3, 6 and 7 are available. The selection 4, the DDD_PULSE is constantly low during free running mode and the option 5 is accordingly constantly high. The DDD_PULSE will only lead a signal in triggered mode.

6.1.8.1   Framewise /TrigOut Option

By default the frame enable signal /FR_EN is attached to the output. The nature of this signal is that it transitions to the low state as soon as the scanning of the image starts. And as soon as the image is finished it toggles back to the high state. So the high state is an indicator when the panel is not actively being scanned which is a good timing to do the x-ray exposure. The good thing about this output signal is that its high pulse width automatically modulates – gets extended – with the increase of the time in between frames when going to slower frame rates or when triggering the camera.
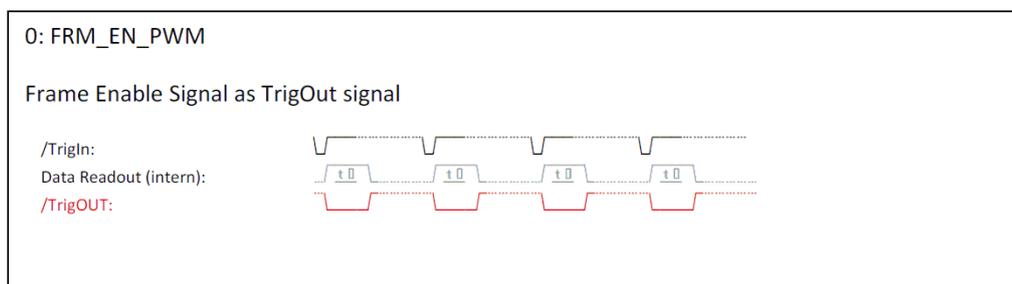
Figure 32:   /Trig Out Option 0: 'Framewise'
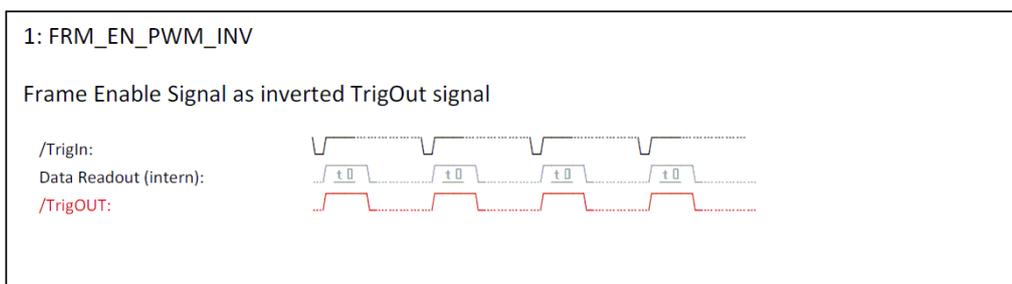
Figure 33:   /Trig Out Option 1: 'Framewise, signal inverted'

6.1.8.2 Exposure Pulse /TrigOut Option

Using the Exposure Pulse /TrigOut option one is capable to expose certain subsequent images within a sequence. By defining the length of this sequence this behaviour will be continiously repeated every *cnt* images. The parameter *i1* and *i2* define which subsequent images are considered for triggering, the length of the /TrigOut edge itself can be adjusted by *d1* and *d2 (length=d2-d1)*. Figure 34 shows the general flow of the 'Exposure Pulse' option, Table 30 describes the individual parameters in detail.
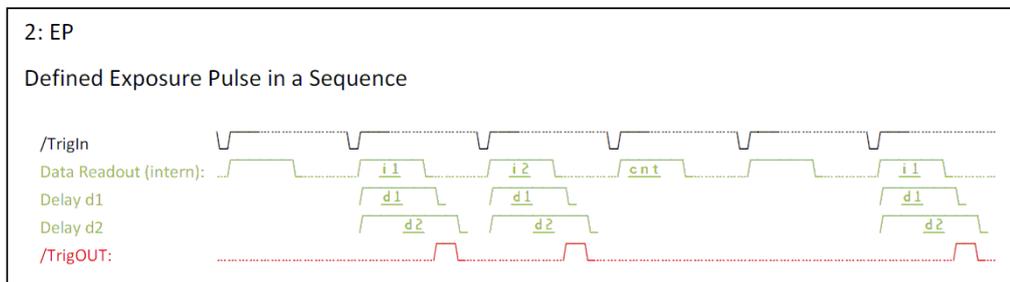
Figure 34: /Trig Out Option 2: 'Exposure Pulse'

Parameter	Description
cnt	The total number of frames of the sequence to be repeated (here: cnt=4)
i1	Index for the first frame within the sequence a trigger out pulse shall be fired (zero-based). (here: i1=1)
i2	Index for the last image within the sequence a trigger out pulse shall be fired. Thus, every frame between *i1* and *i2* incl involves a /TrigOut (zero-based). (here: i2=2)
d1	Delay [in ms] from the begin of a frame that shall be exposed until the trigger out is activated. The delay can be counted from the rising or from the falling edge of the /TrigIn signal. It can be set within the XIS dialog (Menu entry **Options->Detector Options->Select /TriggerOut Signal**) or using the API function Acquisition_SetTriggerOutSignalOptions)
d2	Delay [in ms] from the begin of a frame that shall be exposed until the trigger out is deactivated. The delay can be counted from the rising or from the falling edge of the /TrigIn signal. It can be set within the XIS dialog (Menu entry **Options->Detector Options->Select /TriggerOut Signal**) or using the API function Acquisition_SetTriggerOutSignalOptions)

Table 30: Parameters for 'Exposure Pulse /Trig Out' Option

Note that this option can also be applied in free running mode. In that case the /TrigIn pulse in Figure 34 would be the begin of frame having the selected integration time.

Furthermore there is the possibility to invert the /TrigOut signal:
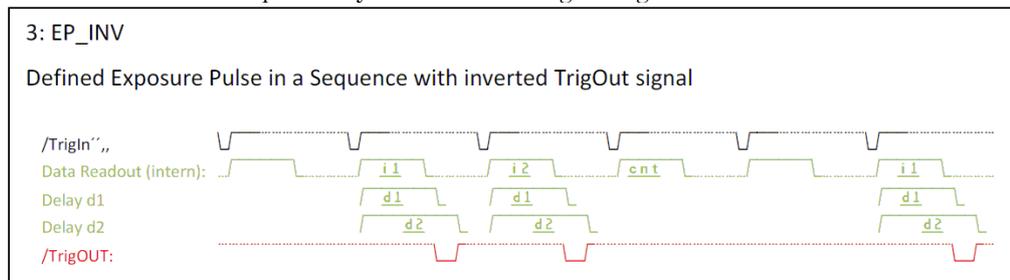
Figure 35: /Trig Out Option 3: 'Exposure Pulse inverted'

6.1.8.3 DDD /TrigOut Option

The detector is running in a "silent" triggered mode, as if in free running mode T0 but without transferring the image data. Eventually the user application triggers the detector. The detector accomplishes the current frame. At the end of the next frame a predefined pause is added allowing the user to radiate the image. A pulse indicating this pause is available at the trigger output. After this pause the detector performs the final image scan and returns to the silent mode. The duration of this pulse is per default set to 400 ms. The DDD Triggerout Option is suggested to be used also in autotrigger-mode
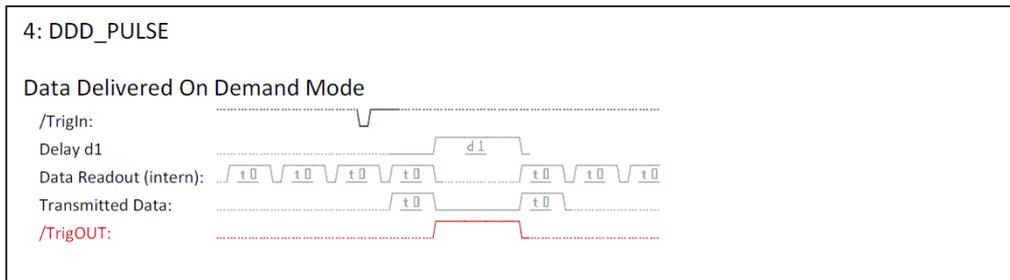
Figure 36: /Trig Out Option 4: 'Data Delivered On Demand'
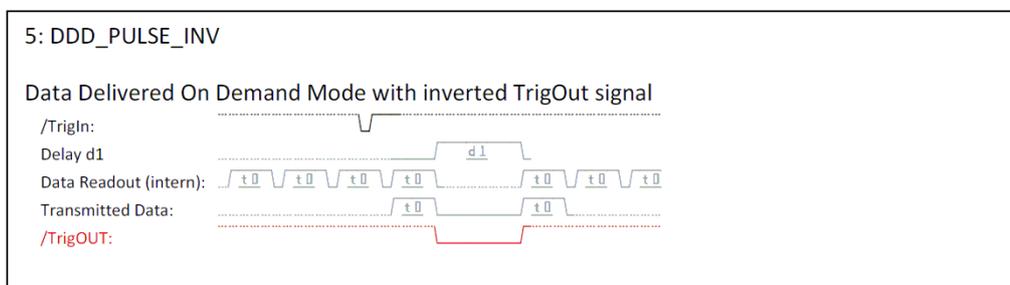
Figure 37: /Trig Out Option 5: 'Data Delivered On Demand inverted'

6.1.8.4 Constant Trigger Output /TrigOut Option

In case no detector-generated TrigOut signal is required, it might be desired to select the state of the output signal being a constant high or constant low. This can be set by the TrigOut options 6 and 7, respectively.
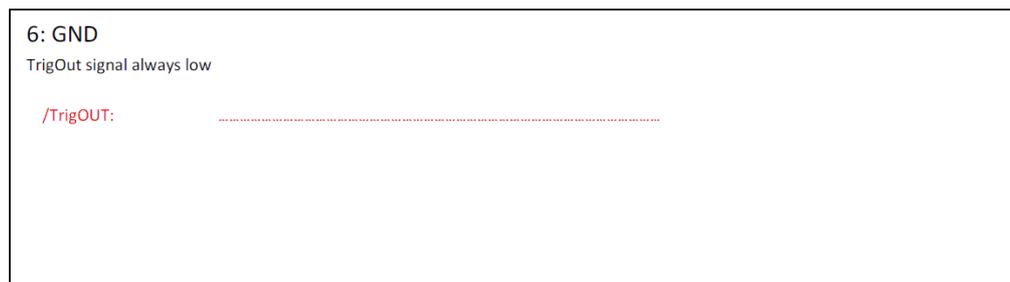
Figure 38: /Trig Out Option 6: 'TrigOut signal on low edge'
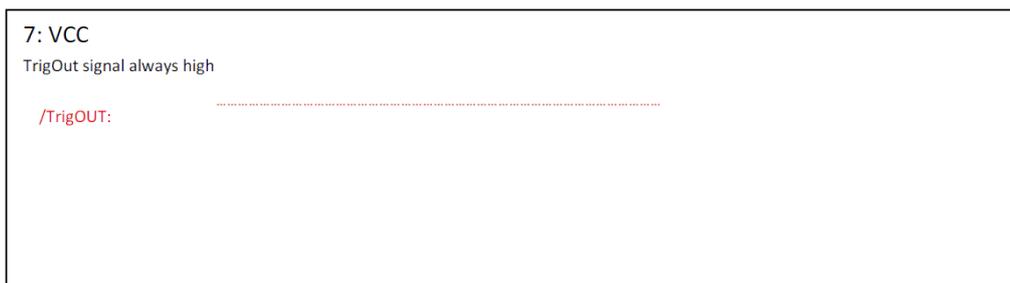
Figure 39: /Trig Out Option 7: 'TrigOut signal on high edge'

6.1.9 External trigger inputs/outputs

The /TRIG IN ,/TRIG OUT, /LN_EN and /FR_EN signals are low active /TTL signals for the XRD 512-400, XRD 1640 AL/AG, XRD 0822 and 1622 detectors and all detectors with the suffix TS.

XRD 08x0 and XRD 16xx detectors provide LVDS Signals. These signals are described in the Table 31. Detailed descriptions of the trigger ports are in the detector manuals.

/FR_ENB	indicates a new frame (detector output)
/FR_SYNC	frame synchronization signal, generated by the frame grabber to externally control the frame rate (detector input)
/LN_ENB	indicates a new line (detector output)
/TRIG_IN	External trigger signal to synchronize the frame rate (detector input)
/TRIG_OUT	The /TRIG_OUT signal indicates the start of a new frame and can be used to synchronise the x-ray source (detector output)

Table 31: Description of the trigger input/output signals

Detector front view

1 – /TRIG IN

2 – /TRIG OUT (Begin of frame)

3 – GND (not used for 0822/1622)

4 – GND (not used for 0822/1622)

5 – GND (required by 0822/1622)

6 – 5 V

7 – NC
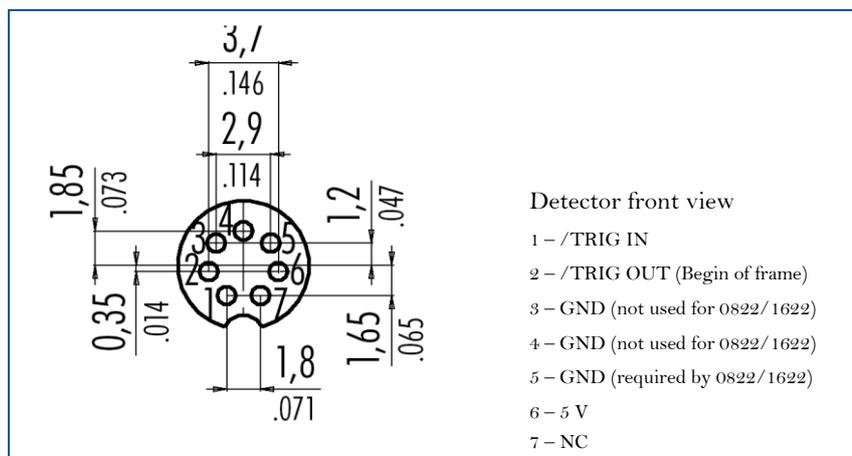
Figure 40   External Trigger inputs/outputs of the XRD 512-400, XRD 1640 AL/AG, XRD 0822 and XRD 1622 detector series

IB – XRD-Interface Bus

1 – /TRIG IN

2 – /FR_EN (Begin of frame)

3 – /FR_SYNC
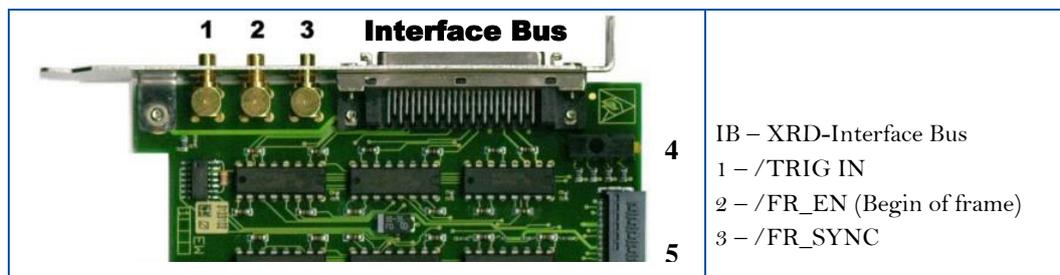
Figure 41: External Trigger inputs/outputs of the XRD-FG Frame Grabber (TTL signals)

Trigger (Front View):

	L1	L2	L3	M	D1	D2	C
Trigger	26	4	22.5	14x1	18	4	12
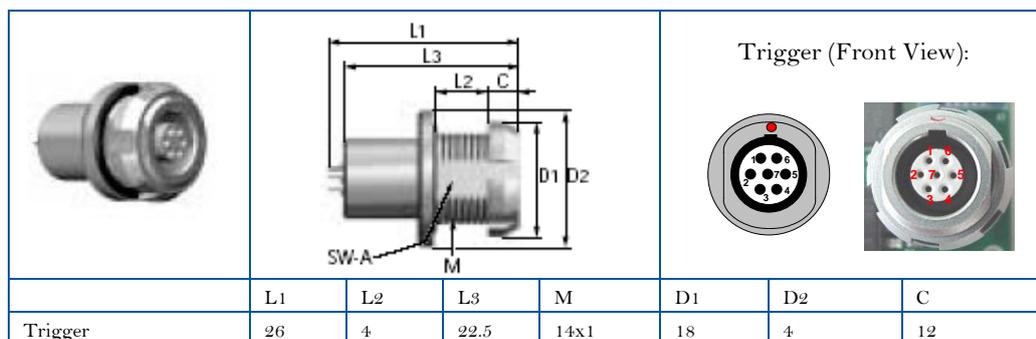
Figure 42: Trigger inputs/outputs of the XRD 08xx yN and XRD 16xx yN detectors

PIN	Colour	Connection
1	Black	TRIGIN_+
2	Brown	TRIG_IN_-
3	Red	TRIGOUT_+
4	Orange	TRIGOUT_-
5	Yellow	FGND
6	Blue	5PF
7	Magenta	

Table 32: PIN assignment of Trigger signal for the XRD 08xx yN and XRD 16xx yN detectors (LVDS Signals)
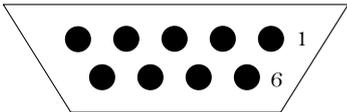
PIN	DESCRIPTION	
1	GP_OUT0+	frame enable
2	GP_OUT1+	frame sync
3	GP_IN0+	trigger in
4	GP_IN1+	na
5	GND	
6	GP_OUT0-	frame enable
7	GP_OUT1-	frame sync
8	GP_IN0-	trigger in
9	GP_IN1-	Na

Table 33: LVDS connector Pin assignment (GP_** use LVDS signalling standard) XRD FG[X][e] Opto

6.1.10 XRpad trigger inputs/outputs

The XRpad provide no trigger connectors. Please use the "Interface Power Unit" to connect external trigger signals to the detector as described in the XRpad Reference Manual or use the software trigger functions.

6.2 Sectional Readout

This paragraph describes the Sectional Readout which is available for detectors with Cameratype 10 and 12. With this function predefined sections of the detector field of view can be selected and combined to one adjacent readout Section. Each of the four sections has a size of 1024 columns and 255 rows (XRD 0822 xP) respectively 128 Rows (XRD 1642 xP).
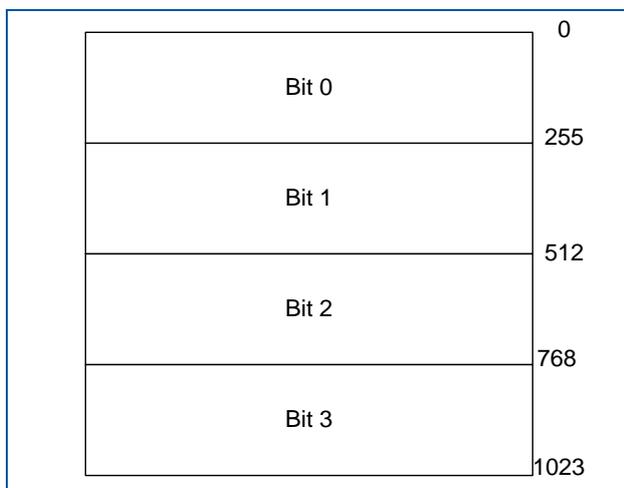
Figure 43   Scheme of the ROI sections XRD 0822 xP

The reduced amount of data according to the selected ROI results in an enhancement of acquisition speed. However, when the Sectional Readout is used the x-ray beam should be collimated to the selected ROI.

Location and size of the region can be set bitwise with the API function Acquisition_SetCameraROI(…) according to Table 59:Bit combination for setting up Sectional Readout (XRD 0822xP/XRD1642xP Detector).

Within the XIS application these values can be set within the dialog 'Select ROI'. This dialog is called by selecting the menu entries **Options/Detector Options/Select Region of Interest (ROI)**.
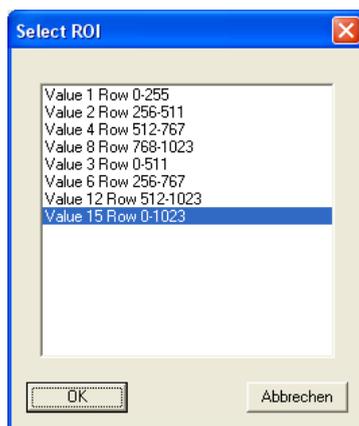
Figure 44 Select Region if Interest dialog (0822AP)

6.3 How to use the detector gain setting

This paragraph describes the change of the detector gain. The **XRD** detectors are supporting different gain settings depending on the detector type (Details in the Detector Manual). When the detector is powered on the XRD runs with the Detector specific default capacity (e.g. 1pF-for AN) . The detector gain will be set by the **XIS** dialog **Options/Detector Options** or by the library function Acquisition_SetCameraGain(*hAcqDesc, 1..5)*. The following table will give an overview of the different gain settings. It is very important for the image quality that the correction files were obtained in the same gain setting as in the current working operating mode.

Selected Value	Capacity
0	0.25pF
1	0.5 pF
2	1 pF
3	2 pF
4	4 pF
5	8 pF

Table 34: Overview of the five Detector Gain Settings

6.4 How to use the detector binning setting

This paragraph describes the change of the detector binning modes. The **XRD detectors** are supporting different binning mode settings depending on the detector type (Details in the Detector manual). When the detector is powered on it runs without binning (default mode). The detector binning will be set by the **XIS** dialog **Options/Detector Options** or by the library function Acquisition_SetCameraBinningMode(*..)*. The following table gives an overview of the different binning modes.

Note: It is very important for the image quality that the correction files are obtained in the same binning mode, gain setting and integration time as for the measurements.

Also be aware that the active mode is always a combination of the actual mode plus the value for averaged or accumulated binning e.g. No binning + Averaged has the value $1 + 256 = 257$.

Active bits	Value	Binning mode	Output image format
0	1	No binning	Square
1	2	2 x 2 binning (averaged or accumulated)	Square
0 + 1	3	4 x 4 binning (averaged or accumulated)	Square
2	4	1 x 2 binning (only accumulated binning)	Rectangular
2 + 1	5	1 x 4 binning (only accumulated binning)	Rectangular
8	256	Averaged binning	
9	512	Accumulated binning	

Table 35: Overview of the detector binning modes

6.5   File format

Each pixel is digitized in 16 bit resolution (2 byte information) and saved as 16 bit unsigned integer for acquired frames. Gain images are stored as unsigned 32 bit integers and there are also double precision floating point data (8 byte) representations supported by the software. All data can be signed or unsigned in general. The data are preceded by a file header of 68 byte and an image header of 32 byte. If we consider a sequence of n images acquired by a 512x512 detector we get the file sizes listed in the table below:

File		Single Image		Sequence	
Header		68	byte	68	byte
Image Header		32	byte	32	byte
Image Data	512x512x2 byte	524288	byte	524288	byte * n

Table 36: Single image and file sequence format

The **file header** allows the reading of the data by any other software module and has the following description:

Information	Description
File header	68 byte
WORD FileType	// File ID (0x7000)
WORD HeaderSize	// Size of this file header in bytes
WORD HeaderVersion	// yy.y
ULONG FileSize	// Size of the whole file in bytes
WORD ImageHeaderSize	// Size of the image header in bytes
WORD ULX, ULY, BRX, BRY	// bounding rectangle of the image
WORD NrOfFrames	// number of frames
WORD Correction	// 0 = none, 1 = Offset, 2 = Offset+Gain
double IntegrationTime	// frame time in microseconds
WORD TypeOfNumbers	// short, long integer, double, signed/unsigned, inverted, fault map, Offset/Gain correction data, pixel correction data
BYTE x[WINRESTSIZE]	// fill up to 68 byte

Table 37: File header description

The file header has a size of 68 byte. It consists of several entries to describe the organization of the stored data. Most of the entries are self-explanatory except the TypeOfNumbers entry, which can have any combination (OR) of the following values:

Fault Map	1 (to be OR-ed with 32)
64 bit double precision floating point number	2
16 bit integer	4
data is signed	8
32 bit integer	32

Table 38: Type of Data

One can get the values of the bounding rectangle by ULX, ULY, BRX, BRY. The number of rows and columns results from the formula:

rows = BRY-ULY+1

columns = BRX-ULX+1

6.6   Description of Hardware Header

The hardware header is transferred by the detector at initialization time of the frame grabber board and at the end of acquisition time.

For a detailed description of the Hardware-Header see Table 55: Description of the ChwHeaderInfo structure and Table 58: Description of the CHwHeaderInfoEx structure

To get all Timings use the Acquisition_GetIntTimes(..)-function while Detector is in FreeRunning-Mode.

6.7   Row types

The following pages describe the available row read out schemes and the corresponding value of RowType (see Hardware Header).

All time values are given in microseconds. In sync mode the timer of the first row starts at synchronization time with a possible time tolerance of 32 nanoseconds. The trigger input is low active. The minimal pulse duration is 1 microsecond and the trigger event is done at the rising edge. This table is obsolete for the latest detector generations (Header ID > 10).

RowType	Row time
0	300
1	300
2	312.5
3	390.625
4	781.25
5	310
6	387.5
7	781.25
8	2500
9	524.0
10	524.0
11	310.0
12	3333
14	2500
15	2500
16	2500
17	2500
18	1536
19	261.0
20	472.0
21	315.9
22	945.0
23	630.0
24	264.0
25	528.0
26	266.0
27	796.0
28	524.0

Table 39: Timing diagram of the row types

6.8   Software Operation Modes (Interrupt vs. Polling)

The following diagrams show the interaction between the PKI Framegrabber Library (XISL) and the customer application.

6.8.1   Interrupt Mode

Interrupts allow the application to wait passively for changes in the acquisition status of the hardware.
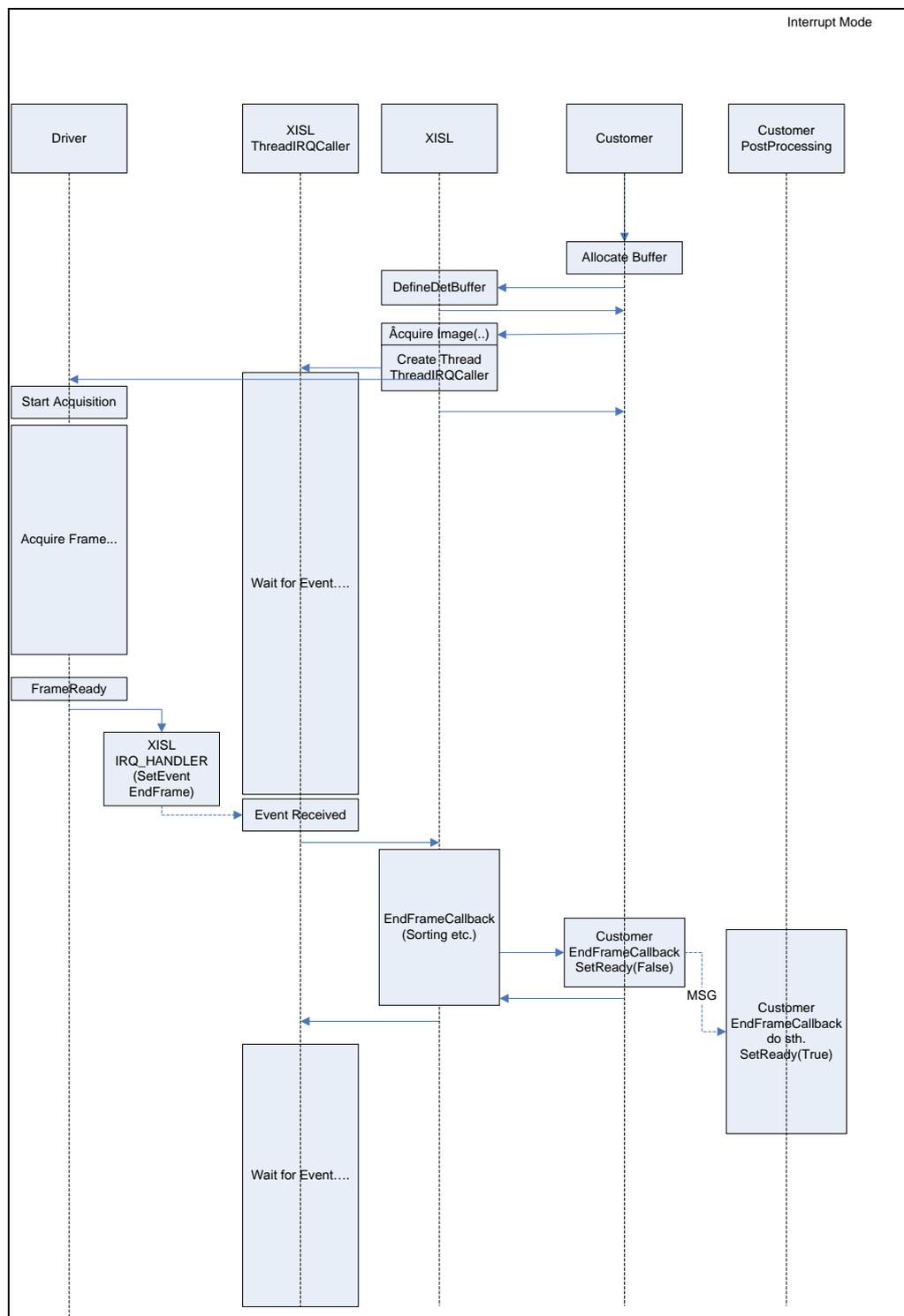
6.8.2 Polling Mode

If no interrupts are enabled the acquisition is running in polling mode. This is a time consuming task because the application actively asks the acquisition driver for its status..
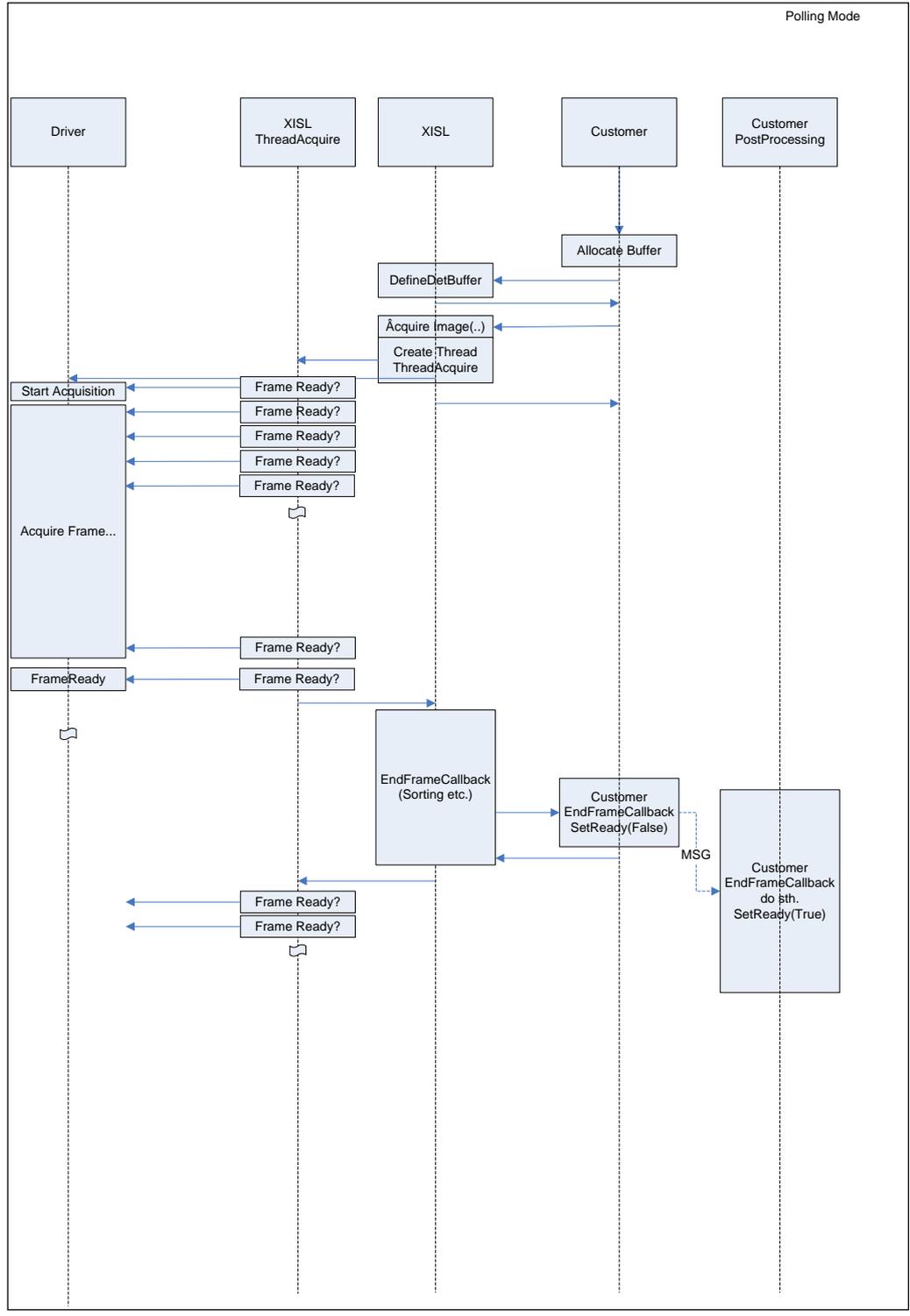
6.9   Technical details for Interrupt sources

There are four interrupt sources:

start of DMA, end of DMA, end of sequence, end of acquisition

These interrupts occur if the acquisition status changes to allow the application to react.

The acquisition mode (Continuous, Single Shot, Sequence) influences the data flow and therefore the acquisition status. The following diagrams illustrate the internal data flow framegrabber and XISL.DLL and the corresponding interrupts.
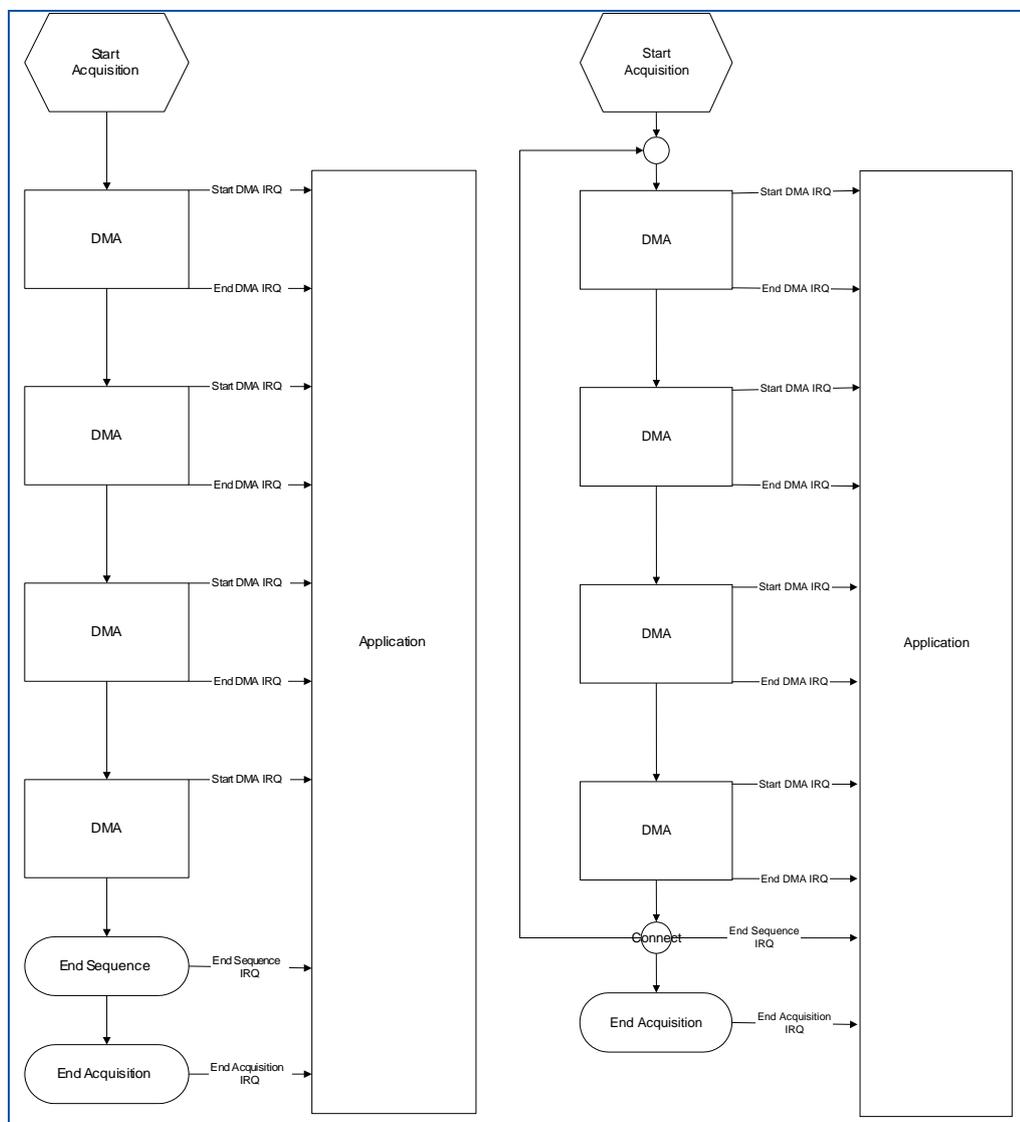
Figure 45: Interrupt sources at a) sequence acquisition mode (4 frames) b) continuous mode.

After the application started the acquisition one interrupt is caused by the beginning of DMA (start DMA interrupt). If the whole frame data is transferred the end of DMA interrupt is fired. Both interrupts show the same behavior in all acquisition modes. What happens if the end of DMA buffer is reached depends on the acquisition mode. In sequence or single shot acquisition mode an end of sequence interrupt occurs and after that an end of acquisition interrupt is caused. In continuous acquisition mode an end of sequence interrupt is caused and the data transfer continuous at start of DMA buffer. If interrupts are enabled, the end of frame, the end of sequence and the end of acquisition interrupts are used. A cancel of acquisition sets the end of frame interrupt.

6.10 Sorting

6.10.1   Sorting schemes overview

Depending on the sensor and detector type the data come in different orders from the detector. The XISL sort the data in an internal buffer with highly optimized routines written in machine code. If the sensor and detector type is unknown the XIS comes up with a detector type dialog at initialization, where the correct sorting has to be entered. The following detector types and sortings are supported:

No Sort (e.g. XRpad, 0822 and 1611, 1622, 1642)	0
XRD 512-400 A1/A2	6
XRD 0840	6
XRD 512-400 E	7
XRD 1640 A	8
XRD 0820	8
XRD 1620 A	8
XRD 1680 A	9
XRD 1620/21 AM/AN	11
XRD 1620/40 AN CS	12

Table 40: Sorting schemes overview

6.10.2 Sorting XRD 512-400 A1/A2 // XRD 0840

The figure demonstrates the read out scheme of the sensor.

The whole sensor is divided into an upper and a lower part. Both parts are electrically separated. The data of every part are transferred by four "read out groups" (ROG). The upper groups scan the sensor columns from left to right. The lower groups scan from right to left. At first the upper groups are transferred and after that the lower ones. The upper groups start their read out from the upper row. The lower ones start read out from the last row. This results in the following Table 41:
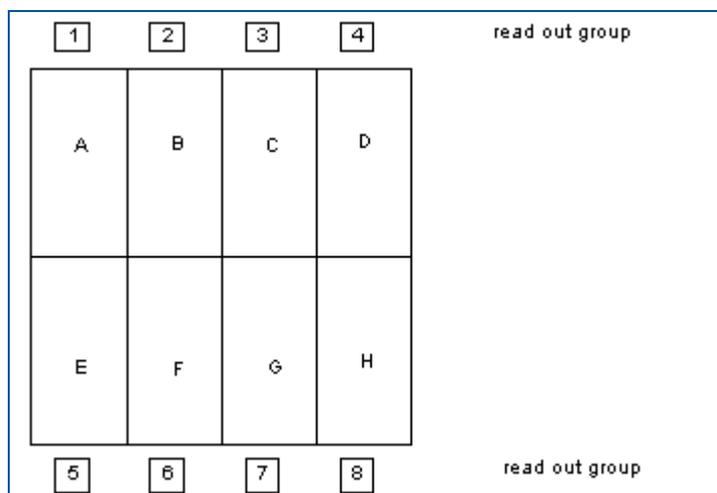
Figure 46: Sorting overview of the XRD 512-400 A1/A2

data stream no.	sensor pixel (row, column)	ROG no.
1	(1,1)	1
2	(1,129)	2
3	(1,257)	3
4	(1,385)	4
5	(512,128)	5
6	(512,256)	6
7	(512,384)	7
8	(512,512)	8
9	(1,2)	1
10	(1,130)	2
…	…	…
262135	(257,258)	7
262136	(257,386)	8
262137	(256,128)	1
262138	(256,256)	2
262139	(256,384)	3
262140	(256,512)	4
262141	(257,1)	5
262142	(257,129)	6
262143	(257,257)	7
262144	(257,385)	8

Table 41: Sorting overview of the XRD 512-400 A1/A2

6.10.3 Sorting XRD 512-400 E

The figure demonstrates the read out scheme of the sensor.

The whole sensor is divided into an upper and a lower part. Both parts are electrically separated. The data of every part are transferred by four "read out groups" (ROG). The upper groups scan the sensor columns from left to right. The lower groups scan from right to left. At first the lower groups are transferred and after that the upper ones. The upper groups start their read out from the upper row. The lower ones start read out from the last row. This results in the following list:
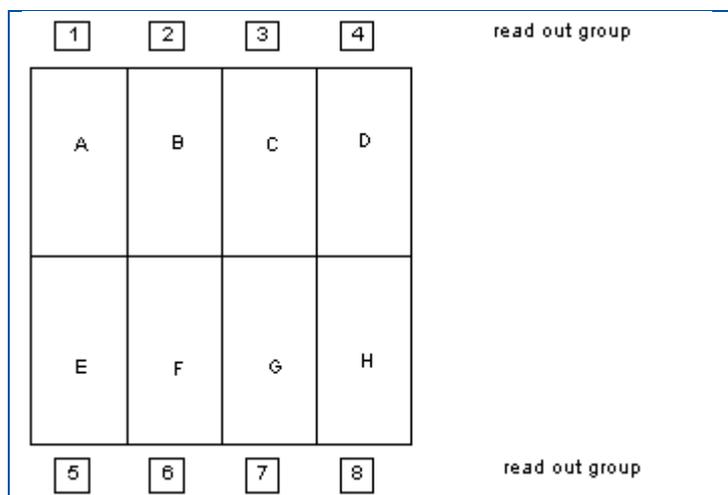
Figure 47: Sorting overview of the XRD 512-400 E

data stream no.	sensor pixel (row, column)	ROG no.
1	(512,128)	5
2	(512,256)	6
3	(512,384)	7
4	(512,512)	8
5	(1,1)	1
6	(1,129)	2
7	(1,257)	3
8	(1,385)	4
9	(512,127)	5
10	(512,255)	6
11	(512,283)	7
12	(512,511)	8
13	(1,2)	1
…	…	…
1022	(1,384)	3
1023	(1,512)	4
1024	(511,128)	5
1025	(511,256)	6
1026	(511,384)	7
1027	(511,512)	8
1028	(2,1)	1
1029	(2,129)	2
…	…	…
262137	(257,1)	5
262138	(257,129)	6
262139	(257,257)	7
262140	(257,385)	8
262141	(256,128)	1
262142	(256,256)	2
262143	(256,384)	3
262144	(256,512)	4

Table 42: Sorting overview of the XRD 512-400 E

6.10.4   Sorting XRD 1640 A // XRD 1620 AJ // XRD 0820

The figure demonstrates the read out scheme of the sensor. The sensors XRD 1620 AJ and XRD 1640 have a similar sorting.

The whole sensor is divided into an upper and a lower part. Both parts are electrically separated. The data of every part are transferred by eight "read out groups" (ROG). Each ROG has 128 channels for the XRD 1640/0820 and 256 for the XRD 1620 AJ. The upper groups scan the sensor columns from left to right. The lower groups scan from right to left. At first the upper groups are transferred and after that the lower ones. The upper groups start their read out from the upper row. The lower ones start read out from the last row.

The following table displays the data stream for XRD 1640:
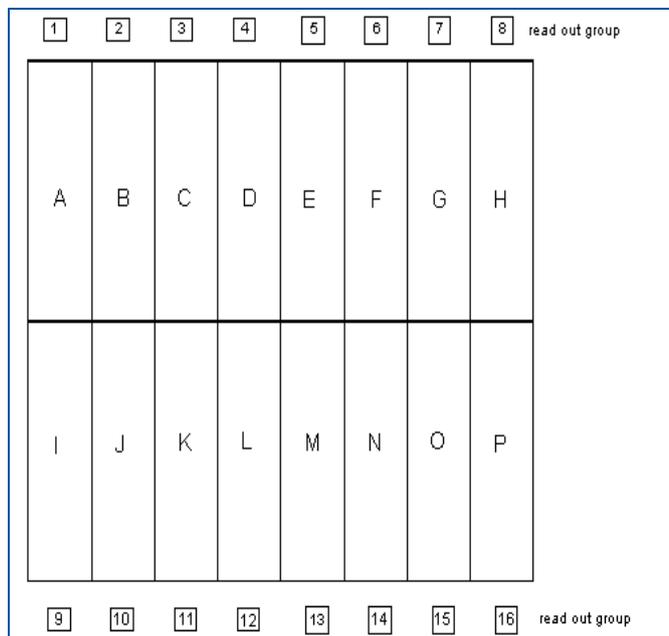
Figure 48: Sorting overview of the XRD 1640 A / 1620 AJ

data stream no.	sensor pixel (row, column)	XRD1620 AJ	ROG no.
1	(1,1)	(1,1)	1
2	(1,129)	(1,257)	2
3	(1,257)	(1513)	3
4	(1,385)	(1,769)	4
5	(1,513)	(1,1025)	5
6	(1,641)	(1,1281)	6
7	(1,769)	(1,1537)	7
8	(1,897)	(1,1793)	8
9	(1024,128)	(2024,256)	9
10	(1024,256)	…..	10
11	(1024,384)		11
12	(1024,512)		12
13	(1024,640)		13
14	(1024,768)		14
15	(1024,896)		15
16	(1024,1024)		16
17	(1,2)		1
18	(1,130)		2
…	…		…
1048570	(513,129)		10
1048571	(513,257)		11
1048572	(513,385)		12
1048573	(513,513)		13
1048574	(513,641)		14
1048575	(513,769)		15
1048576	(513,897)		16

Table 43: Sorting overview of the XRD 1640 A / 1620 AJ

6.10.5   Sorting XRD 1620 /21 AM/AN

The figure demonstrates the read out scheme of the sensor.

The whole sensor is divided into an upper and a lower part. Both parts are electrically separated. The data of every part are transferred by 16 "read out groups" (ROG). Each ROG has 128 channels. The upper groups scan the sensor columns from left to right. The lower groups scan from right to left. At first the upper groups are transferred and after that the lower ones. The upper groups start their read out from the upper row. The lower ones start read out from the last row.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
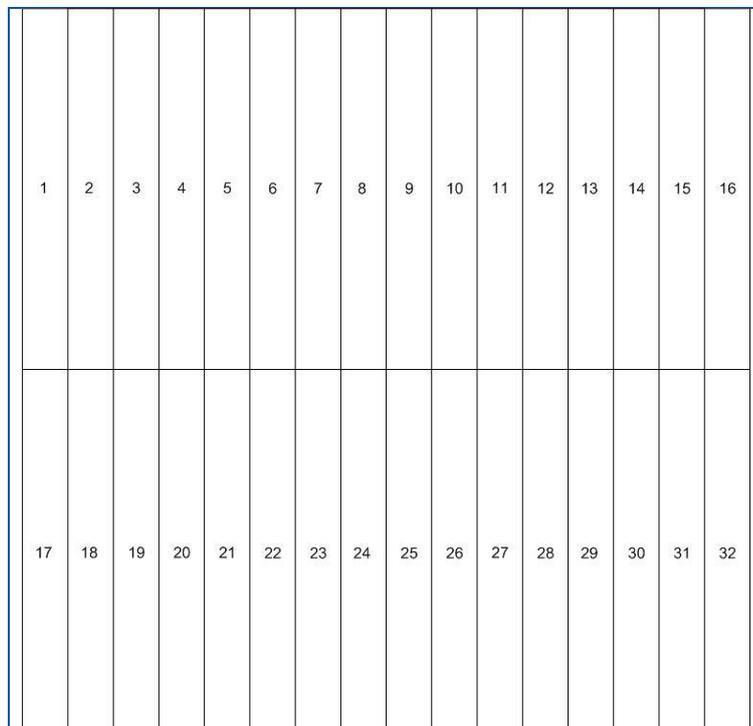
Figure 49: Sorting overview of the XRD 1620 AN / 1621 AN

The following table displays the data stream for XRD 1620 AM:

data stream no.	sensor pixel (row, column)	ROG no.
1	(1,1)	1
2	(1,129)	2
3	(1,257)	3
4	(1,385)	4
5	(1,513)	5
6	…	
15	(1,1793)	15
16	(1,1921)	16
17	(2048,128)	17
18	(2048,256)	18
19	(2048,384)	19
20	(2048,512)	20
…	…	…

Table 44: Sorting overview of the XRD 1620/21 AN//AM

6.10.6 Sorting XRD 1620 / 40 AN CS

The figure demonstrates the read out scheme of the sensor.

The data are transferred by 16 "read out groups" (ROG). Each ROG has 128 channels (64 for 1640). The groups scan from right to left. The rows start read out from the last row.
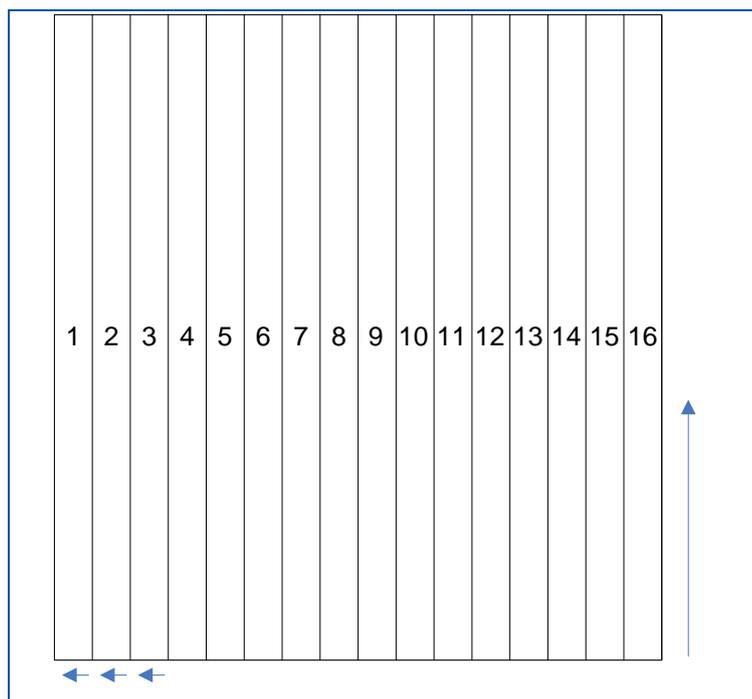
Figure 50: Sorting overview of the XRD 1620/40 AN CS

The following table displays the data stream for XRD 1620 AN CS:

data stream no.	sensor pixel (row, column) 1620	sensor pixel (row, column) 1640	ROG no.
1	(2048,128)	(1024,64)	1
2	(2048,256)	(1024,128)	2
3	(2048,384)	(1024,192)	3
…	…	…	…
16	(2048,2048)	(1024,1024)	16
17	(2048,127)	(1024,63)	1
18	(2048,255)	(1024,127)	2
19	(2048,384)	(1024,191)	3
…	…	…	…
32	(2048,2047)	(1024,1023)	16
…	…	…	…

Table 45: sorting overview of the XRD 1620/40 AN CS

6.10.7 Sorting XRpad, XRD 0822, XRD 1622, XRD 1642

The image data comes already sorted from the detector. No additional sorting is necessary.

Inernal readout sceme:

The data are transferred by 8 (0822, 1642) /16 (1622) /56 (XRpad) "read out groups" (ROG).

Each ROG has 128 / XRpad 64/63 channels. The groups scan from right to left. The rows start read out from the last row.

The following Figure 51 shows the read out scheme and the Table 46 displays the data stream for **XRD 0822**:
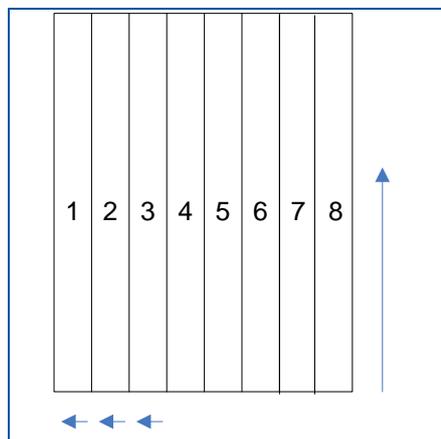
Figure 51   Sorting scheme of the XRD 0822

data stream no.	sensor pixel (row, column)	ROG no.
1	(1024,128)	1
2	(1024,256)	2
3	(1024,384)	3
…	…	…
8	(1024,1024)	8
9	(1024,127)	1
10	(1024,255)	2
11	(1024,384)	3
…	…	…
16	(1024,1023)	8

Table 46:   Sorting scheme of the XRD 0822

6.11 Detector Options Overview

The table provides information about the capabilities of the different detector models. The soft- and firmware identify the functionality of a detector by the camera type. This number can be retrieved within the header data for recent detectors which support Header ID >= 14. Earlier detector models with Header ID < 14 do not support this feature.

The features in the columns *Trigger Modes* and *Trigger Output* are implemented in firmware as listed. The software support is tested for detectors with camera type >7.

Detector Model	Data Interface	Header ID	Camera Type	Bit Depth	Frame rate [fps] 1x1	2x2	4x4	1x2	1x4	Modes	Gain [pF] 0,25	0,5	1	2	4	8	Trigger Modes	Trigger Output	Sectional Readout
XRD 0820 xN	HIIB	14	1	16	7,5	15	-	-	-	AVG	x	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 0820 xN ES	HIIB	14	2	16	15	30	-	-	-	SUM, AVG	x	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 0820 BE ES	HIIB	14	2	14	15	30	-	-	-	SUM, AVG	x	-	-	x	-	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 0840 xN	HIIB	13	-	16	15	-	-	-	-		-	x	x	x	x	-	FW, LW+	FW, FW-, EP, EP-, Constant	-
XRD 0840 BE	HIIB	13	-	14	30	-	-	-	-		-	-	x	-	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 1620 xN	HIIB	13	-	16	3,5	-	-	-	-		-	x	x	x	x	x	FW	FW	-
XRD 1620 xN CS	HIIB	14	2	16	3,75	7,5	-	-	-	SUM, AVG	-	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 1640 xN ES	HIIB	13	-	16	15	-	-	-	-		-	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 1640 xN ES	HIIB	14	7	16	15	-	-	-	-	FW	-	-	x	x	-	x	FW	FW, FW-, EP, EP-, Constant	-
XRD 1640 xN ES	HIIB	13	-	16	15	-	-	-	-		-	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 1640 xN CS	HIIB	13	-	16	7,5	-	-	-	-	AVG	-	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 1621 xN	OPT	14	1	16	7,5	15	-	-	-	AVG	x	x	x	x	x	x	FW	FW, FW-, EP, EP-, Constant	-
XRD 1641 xN	OPT	14	1	16	15	30	-	-	-		x	x	x	x	x	x	FW	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 1621 xN ES	OPT	14	2	16	15	30	-	-	-	SUM, AVG	x	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 0822 xO	GbIF	14	8	14	15	30	-	-	-	SUM, AVG	-	-	x	-	x	-	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 0822 xP	GbIF	14	10	16	25	50	100	50	100	SUM, AVG	x	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	1024x512 1024x256
XRD 1622 xO	GbIF	14	9	14	1	-	-	-	-	SUM, AVG	-	-	x	-	x	-	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 1622 xP	GbIF	14	11	16	1	4	-	2	-	SUM, AVG	x	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 1642 xP	GbIF	14	11	16	15	30	-	30	-	SUM, AVG	x	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRD 1611 xP	OPT	14	15	16	3,75	7,5	15	7,5	15	SUM, AVG	x	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	1024x512 1024x256 1024x128
XRD 1642 xP	GbIF	14	12	16	15	30	-	30	-	SUM, AVG	x	x	x	x	x	x	FW, DDD, LW+	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRpad 4336	GbIF / WLAN	14	13	14/16	Refer to the XRpad reference manual for selectable detector options (Gain, Binning and Frame Rate)												FW, DDD, LW+, Auto	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRpad 4343 F	GbIF	14	13	16													FW, DDD, LW+, Auto	FW, FW-, EP, EP-, DDD, DDD-, Constant	-
XRpad 4343	GbIF / WLAN	14	13	16													FW, DDD, LW+, Auto	FW, FW-, EP, EP-, DDD, DDD-, Constant	-

Table 47: Detector Options Overview

Definitions

FW	Frame Wise
PW-	Frame Wise, Pulse Inverted
DDD	Data Delivered on Demand
DDD-	DDD - Pulse Inverted
LW	Line Wise w/ fixed lines
LW+	Line Wise with Start Stop
EP	Exposure Pulse
EP-	Exposure Pulse Inverted
TL	Trigger Loss
TM	Trigger Mode
TS	Trigger Soure 1=Intern 0=Extern
SM	Service Mode
HIIB	Copper based interface (Framegrabber required)
OPT	Optical data interface (Framegrabber required)
GbIF	Gigabit Ethernet data interface

7 X-Ray Imaging Software Library SDK

The X-Ray Imaging Software Library (XISL) makes it easy to integrate flat panel functionality into the user application. The SDK for 32 as well as for 64bit can be selected as an installation option during the XIS Setup.
The SDK comprises a selection of Standard C-functions to allow the user to put into action the desired mode of operation. The SDK provides functions for image acquisition and detector control such as triggering, binning, setting gain and more.
Please ensure to include the adequate acq.h (32 or 64bit) into your project. For 32bit you can find this file in the folder [XIS]\SDK32, the 64bit version is located in [XIS]\SDK64.

7.1 X-Ray Imaging Library Overview

The X-Ray Imaging Software Library provides the basic functionality to acquire and correct XRD/RID data. To get an impression of the tasks you have to implement to acquire images by help of the XISL look at XISL demo. This demo is also a part of the XIS setup.

Group	Function Name	Description
Init	Acquisition Descriptor	Data acquisition handle
	Acquisition_EnumSensors	Enumerates all connected sensors
	Acquisition_GetNextSensor	Iterates through all recognized sensors.
	Acquisition_Init	Initializes driver and frame grabber.
	Acquisition_GetConfiguration	Retrieves the current configuration setting of the XISL.
	Acquisition_GetIntTimes	Routine to detect the actual frame time automatically.
	Acquisition_GetHwHeaderInfo	Returns the contents of the detector's hardware header in an info structure.
	Acquisition_GetHwHeaderInfoEx	Acquire Header and Extended Header if available
	Acquisition_GetCommChannel	Retrieves the connected ID of the connected data interface
	Acquisition_SetCallbacksAndMessages	Defines callback functions to react on acquisition status changes
	Acquisition_Close	Closes drivers and hardware of the specified detector
	Acquisition_CloseAll	Closes drivers and hardware for all open detectors
	Acquisition_wpe_GetVersion	This function returns the version of the used wpe200 library
Init GbIF	Acquisition_GbIF_Init	Initializes driver and GbIF device.
	Acquisition_GbIF_CheckNetworkSpeed	Retrieves the proposed Detector Timing and Packetdelay for the current system and network configuratio
	Acquisition_GbIF_GetDeviceCnt	Retrieves the number of GBIF devices found by network broadcast
	Acquisition_GbIF_GetDeviceList	Retrieves additional info for each found device
	Acquisition_GbIF_GetDevice	Retrieves single devices by MAC-Address, IP-Address or device name
	Acquisition_GbIF_GetDeviceParams	Retrieves device-parameters of a GigE Detector
	Acquisition_GbIF_SetConnectionSettings	Set boot parameter of network connection
	Acquisition_GbIF_GetConnectionSettings	Retrieves boot parameter of network connection
	Acquisition_GbIF_SetPacketDelay	Set Inter-Packet Delay value
	Acquisition_GbIF_GetPacketDelay	Retrieves Inter-Packet Delay value
	Acquisition_GbIF_ForceIP	Assign temporary IP for initial connection
	Acquisition_GbIF_GetDetectorProperties	Retrieves the detector production data
	Acquisition_GbIF_GetFilterDrvState	Retrieves the status of the GbIF Filter Driver
Init XRpad	Acquisition_wpe_getAvailableSystems	This function sends a networkbroadcast and retrieves all available XRpad detectors in the Network
	Acquisition_wpe_GetNetworkConfigs	This function retieves the Network settings of a specific Detector defined by its ip address
	Acquisition_wpe_ActivateNetworkConfig	This function activates the selecte network configuration of the detector
	Acquisition_wpe_ReadCameraRegisters	This function retrieves the Statusregister of a specific Detector definded by its ip address
Acquire / correct	Acquisition_Acquire_Image	Acquires images from the detector.
	Acquisition_Acquire_GainImage	Acquires a gain correction image.
	Acquisition_Acquire_OffsetImage	Acquires an offset correction image.
	Acquisition_DoOffsetCorrection	Performs an offset correction on an acquired image.
	Acquisition_DoOffsetGainCorrection	Performs offset and gain correction on an acquired image.
	Acquisition_DoPixelCorrection	Performs a pixel correction on an acquired image.
	Acquisition_DefineDestBuffers	Definition of the destination buffers for image capturing.
	Acquisition_CreatePixelMap	Creates a list for a mean correction of defective pixels.
	Acquisition_GetLatestFrameHeader	Retrieve Last Frame Header and Extended Header if available

	Acquisition_Acquire_Image_PreloadCorr	Acquire Image w/o setting correction data
	Acquisition_Acquire_GainImage_PreloadCorr	Acquire Gain Image w/o setting correction data
	Acquisition_Acquire_OffsetImage_PreloadCorr	Acquire Offset Image w/o setting correction data
	Acquisition_ Acquire_GainImage_EX_ROI	Acquire Gain Image with defined Region of interest
	Acquisition_Acquire_GainImage_Ex_ROI_PreloadCorr	Acquire Gain Image (ROI) w/o setting correction data
	Acquisition_CreateGainMap	Create a List of Median values for the Gain-Sequence correction
	Acquisition_Acquire_Image_Ex	Acquires images from the detector.
	Acquisition_SetCorrData	This function switches the correction buffers
	Acquisition_GetCorrData	This function retrieves the correction buffers
	Acquisition_SetCorrData_Ex	This function switches the correction buffers during a running acquisition.
	Acquisition_GetCorrData_Ex	This function retrieves the correction buffers during a running acquisition
	Acquisition_DoOffsetGainCorrection_Ex	Performs offset and gain-sequence correction on an acquired image.
	Acquisition_SetAcqData	Sets a 32 bit value / pointer that can be extracted by Acqusition_GetAcqData during acquisition time.
	Acquisition_Abort	Aborts a running acquisition.
	Acquisition_AbortCurrentFrame	Aborts the transmission of the current frame and immediately starts a new transfer.
Detector settings	Acquisition_SetCameraMode	Allows setting of detector frame time.
	Acquisition_SetCameraGain	Set Detector Gain (not available for all detectors)
	Acquisition_SetCameraBinningMode	Set binning Mode (not available for all detectors)
	Acquisition_GetCameraBinningMode	Retrieve actual binning Mode.
	Acquisition_SetCameraTriggerMode	Set trigger Mode (not available for all detectors)
	Acquisition_GetCameraTriggerMode	Retrieve actual trigger Mode.
	Acquisition_ResetFrameCnt	Reset detector Frame counter (not for all detectors)
	Acquisition_SetFrameSyncTimeMode	Set Parameters for 'Data Delivered on Demand' triggermode
	Acquisition_SetTimerSync	Set Internal Trigger Time
	Acquisition_SetFrameSyncMode	This function sets the frame sync mode of the detector (free_running, internal-, external-, auto- or soft trigger)
	Acquisition_SetFrameSync	This function supplies a software synchronization signal to the detector
	Acquisition_SetTriggerOutSignalOptions	This function defines the behavior of the '/TrigOut' - signal of the detector trigger connector.
	Acquisition_SetCameraROI	This function enables a selectable Regions of Interest for Readout.
	Acquisition_GetCameraROI	This function returnes the current activated Region of Interest for Readout.
	Acquisition_ActivateServiceMode	This function activates the Service Mode.
	Acquisition_GetFTPFile	This function retrieves a file from the SD-Card to a given buffer
	Acquisition_FreeFTPFileBuffer	This function frees the buffer retrieved by Acquisition_GetFTPFile
	Acquisition_SetFTPFile	This function stores a databuffer on the sd card
	Acquisition_SetRotationAngle	This function selects the image rotation angle on optical frame-grabbers for images up to 2048x2048 pixels
CallBack	Acquisition_GetReady	This function checks if the passed Acquisition Descriptor is valid.
	Acquisition_SetReady	Informs the XISL that the user image processing is ready.
	Acquisition_IsAcquiringData	Checks if an Acquisition Thread is running
	Acquisition_GetWinHandle	Returns the handle of the current acquisition window.
	Acquisition_GetAcqData	Extracts a 32 bit value / pointer at acquisition time that was set by Acqusition_SetAcqData.
	Acquisition_GetActFrame	Retrieves the nr of the current frame in the ring buffer.
	Acquisition_GetTriggerOutStatus	This function retrieves the triggerout status from detector
Error Handling	Acquisition_GetErrorCode	Returns extended information if an error occurred.
	Acquisition_wpe_GetErrorCode	This function retrieves the last error code from the wpe200.dll
	Acquisition_wpe_GetErrorCodeEx	This function retrieves the last error code as a string from the wpe200.dll

Table 48: Overview of the X-Ray Imaging Software Library

7.2   XISL Functions

The following API descrition is valid for:

XISL version 3-3-27-32 or higher (32bit API)

XISL version 3-3-27-64 or higher (64bit API)

7.2.1   Acquisition Descriptor

AcquisitionDesc defines a data structure that is used by all functions of XISL. It contains all required parameters for the acquisition. Access to the data fields is only possible via the XISL API functions. **HACQDESC** defines a pointer to the acquisition descriptor. A valid **HACQDESC** pointer is returned by Acquisition_Init and all allocated resources are freed by Acquisition_Close. After a call of Acquisition_Close the acquisition descriptor pointer is invalid.

7.2.2   Acquisition_EnumSensors

This function enumerates all currently connected sensors. All recognized sensors are initialized automatically. To get the **HACQDESC** of every sensor, use Acquisition_GetNextSensor. For a programming example see the initialization part of the XISL demonstration.
Note: In case of GbIF Sensors only directly connected sensors (point-to-point) are initialized automatically (i.e. the standard-gateway is equal to zero) .

UINT WINAPI Acquisition_EnumSensors(
      UINT *pdwNumSensors,
      BOOL bEnableIRQ,
      BOOL bInitAlways
      );
PdwNumSensor
      Address of a 4 byte integer that receives the number of recognized sensors.
bEnableIRQ
      If you want to run the acquisition in polling mode set this parameter to zero. If you want
      to enable hardware interrupts set the parameter to one.
bInitAlways
      If this parameter is TRUE the XISL is capturing all communication port regardless if
      this port is already opened by other processes running on the system. The use of this
      option is only recommended in debug versions of your applications because it is not
      possible to free all system resources allocated by another process.
Return Value
      If the initialization is successful zero is returned, otherwise the return value is greater.
      Call Acquisition_GetErrorCode to get extended information.

7.2.3   Acquisition_GetNextSensor

You can use this function to iterate through all recognized sensors in the system.
for a programming example see the initialization part of the XISL demonstration.
UINT WINAPI Acquisition_GetNextSensor(
      ACQDESCPOS *Pos,
      HACQDESC *hAcqDesc
      );
Pos

      Pointer to an unsigned 4 byte integer (32bit library) that receives information's that are
      needed for subsequent calls of this function. To receive the acquisition descriptor
      (HACQDESC) for the first recognized sensor set Pos to NULL.
      In case of the 64bit library ACQDESCPOS is a void pointer like defined in acq.h of the
      driver SDK.
hAcqDesc
      Handle of a structure that contains all needed parameters for acquisition (HACQDESC).
      If you call Acquisition_Init the first time set hAcqDesc to NULL, in subsequent calls use
      the former returned value.

Return Value

> If the initialization is successful zero is returned, otherwise the return value is greater.
> Call Acquisition_GetErrorCode to get extended information.

7.2.4　　　Acquisition_Init

The **Acquisition_Init** function initializes the frame grabber board and the corresponding driver. It enables desired hardware interrupts, prepares acquisition threads, defines callback functions to react on acquisition status changes and tests for sufficient memory space for DMA (direct memory access).

UINT WINAPI Acquisition_Init(
  HACQDESC *phAcqDesc,
  DWORD dwBoardType,
  int nChannelNr,
  BOOL bEnableIRQ,
  UINT Rows,
  UINT Columns,
  UINT dwSortFlags,
  BOOL bSelfInit,
  BOOL bInitAlways
  );

phAcqDesc

> Handle of a structure that contains all needed parameters for acquisition (HACQDESC).
> If you call Acquisition_Init the first time set hAcqDesc to NULL, in subsequent calls use the former returned value.

dwBoardType

> This parameter defines on which communication device the sensor is located. Only one type of frame grabber can be used at the same time.

dwBoardType can have the following values:

symbolic name	numeric value	Meaning
HIS_BOARD_TYPE_ELTEC	1	The communication interface to the detector is an XRD-FG or XRD-FGe frame grabber.
HIS_BOARD_TYPE_ELTEC_XRD_FGX	8	The communication interface to the detector is an XRD-FGX frame grabber.
HIS_BOARD_TYPE_ELTEC_XRD_FGE_Opto	16	The communication interface to the detector is an XRD-FGe Opto frame grabber.
HIS_BOARD_TYPE_ELTEC_GbIF	32	The detector communicates over GigabitEthernet with the host PC. (For GbIF and XRpad Detectors **Acquisition_GbIF_Init** must be used)
HIS_BOARD_TYPE_ELTEC_EMBEDDED	96	The detector communicates over GigabitEthernet or WLAN with the host PC. (For GbIF and XRpad Detectors **Acquisition_GbIF_Init** must be used) This value is an ored combination of HIS_BOARD_TYPE_ELTEC_GbIF and HIS_BOARD_TYPE_ELTEC_WPE showing that extended Functionality to the GbIF is available. The value will be set automatically after initialization and can be retrieved by Acquisition_GetCommChannel(..)

Table 49: Supported interfaces and channel types

dwChannelNr

> This parameter defines the device number. Its possible values depend from *dwBoardType* and the number of the installed components. For instance if you installed 2 frame grabber boards and you want to acquire data from that one, on that the hardware board selector is set to three, set d*wChannelNr* equal to 3.

bEnableIRQ

> If you want to run the acquisition in polling mode set this parameter to zero. If you want to enable hardware interrupts set the parameter to one.

Rows, Columns

> Number of sensor columns and rows.

dwSortFlags

> Depending on the sensor different sorting schemes are needed because the data come in incorrect order from the detector. dwSortFlags can be one of the following values:

HIS_SORT_NOSORT (0x0)	no sorting / XRpad series / 0822 / 1622 / 1642 / 1611
HIS_SORT_QUAD (0x1)	RID128
HIS_SORT_COLUMN (0x2	RID256
HIS_SORT_COLUMNQUAD (0x3)	RID128-400
HIS_SORT_QUAD_INVERSE (0x4)	RID1024-100
HID_SORT_QUAD_TILE (0x5)	RID512-400 A0
HIS_SORT_QUAD_TILE_INVERSE (0x6)	XRD512-400 A1/A2 XRD 0840
HIS_SORT_QUAD_TILE_INVERSE_SCRAMBLE (0x7)	XRD 512-400 E
HIS_SORT_OCT_TITLE_INVERSE (0x8)	XRD 1640 A , XRD 1620 A , XRD 0820
HIS_SORT_HEX_TILE_INVERSE (11)	XRD 1620/21 AM/AN
HIS_SORT_HEX_CS(12)	XRD 1620/40 AN CS

Table 50: Sorting Flags

> The sorting is done automatically by XISL during acquisition. The sorting routines are written in machine code and are therefore very fast.

bSelfInit

> If *bSelfInit* is set to true the function retrieves the detector parameters (Rows, Columns,SortFlags) automatically.
> If *bSelfInit* is set to false the configuration parameters supplied by *Rows, Columns, dwSortFlags* are used.

bInitAlways

> If this parameter is TRUE the XISL is capturing the requested communication port regardless if this port is already opened by other processes running on the system. The use of this option is only recommended in debug versions of your applications because it is not possible to free all system resources allocated by another process.

Return Value

> If the initialization is successful zero is returned, otherwise the return value is greater. Call Acquisition_GetErrorCode to get extended information.

7.2.5 Acquisition_GetCommChannel

This function returns the type of the communication device that is used to transfer data from the detector into the PC RAM.

UINT WINAPI Acquisition_GetCommChannel(

      HACQDESC *hAcqDesc,*

      UINT *pdwChannelType,

      int *pnChannelNr

      );

hAcqDesc

      Pointer to HACQDESC.

pdwChannelType

      Address of a 4 byte integer that receives an id of the currently open communication
      device. Currently supported devices and their corresponding id's are:

Symbolic name	id	Description
HIS_BOARD_TYPE_NOONE	0	no device (not valid)
HIS_BOARD_TYPE_ELTEC	1	XRD-FG or XRD-FGe Frame Grabber
HIS_BOARD_TYPE_ELTEC_XRD_FGX	8	XRD-FGX frame grabber.
HIS_BOARD_TYPE_ELTEC_XRD_FGE_Opto	16	XRD-FGe Opto
HIS_BOARD_TYPE_ELTEC_GbIF	32	GigabitEthernet
HIS_BOARD_TYPE_ELTEC_EMBEDDED	96	GigabitEthernet or WLAN interface GbIF functions and functions supporting the XRpad are available

Table 51: Supported Frame Grabber device and their channel type

pnChannelNr

      Address of a 4 byte integer that receives the number of communication channel. If the
      above mentioned communication device is a frame grabber this number is unique to
      identify the grabber if more than one grabber of one type is installed on the system. (see
      frame grabber installation description). If the communication device is an RS232 interface
      then this number contains the COM port number.

7.2.6 Acquisition_DefineDestBuffers

This function defines the pointers of the destination buffer for Acquisition_Acquire_Image. The data are written into this buffer after sorting. The buffer must have a proper size depending on acquisition mode. To acquire one image the buffer must have the size sensor rows * sensor columns *2. To acquire a sequence the buffer must have the size sensor rows * sensor columns *2 * frames. In the case of continuous acquisition the buffer must have the size sensor rows * sensor columns *2 * frames of the ring buffer.

UINT WINAPI Acquisition_DefineDestBuffers(

      HACQDESC hAcqDesc,

      unsigned short *pProcessedData,

      UINT nFrames,

      UINT nRows,

      UINT nColumns

      );

hAcqDesc

      Pointer to HACQDESC.

pProcessedData

      Pointer to the destination buffer.

nFrames

      Number of frames of the destination buffer. It must be greater than zero.

nRows, nColumns

      Number of rows and columns of the destination buffer. If these numbers are not suitable
      to the sensor the function return with an error code. If you need extended information
      call Acquisition_GetErrorCode.

Return Value

Zero if function is successful, otherwise with a greater value.

7.2.7 Acquisition_SetCallbacksAndMessages

The **Acquisition_SetCallbacksAndMessages** function defines callback functions to react on acquisition status changes. For a programming example see the initialization part of the XISL demonstration.

UINT WINAPI Acquisition_SetCallbacksAndMessages(
    HACQDESC hAcqDesc,
    HWND hWnd,
    UINT dwErrorMsg,
    UINT dwLoosingFramesMsg,
    void (CALLBACK *lpfnEndFrameCallback)(HACQDESC),
    void (CALLBACK *lpfnEndAcqCallback)(HACQDESC)
    );

hAcqDesc

    Handle of a structure that contains all needed parameters for acquisition (HACQDESC). If you call Acquisition_Init the first time set hAcqDesc to NULL, in subsequent calls use the former returned value.

hWnd

    If the XISL recognizes an end of DMA transfer and it is ready with sorting, it checks if the application called Acquisition_SetReady after redrawing. If the application did not call the function, an user defined message (dwLoosingFramesMsg) is posted to hWnd for further handling. If an error occurred during acquisition also a user defined message (dwErrorMsg) is posted to hWnd.

dwErrorMsg

    Defines a user message that is posted to hWnd if an error occurs during acquisition.

dwLoosingFramesMsg

    Defines a user message that is posted to hWnd if Acquisition_SetReady wasn't called by the application at the end of sorting.

lpfnEndFrameCallback

    Defines a function pointer that is called after the XISL did the sorting. The prototype for the function is given by:

void CALLBACK OnEndFrameCallback(HACQDESC hAcqDesc);

    In this routine you can do corrections, on-line image processing and redrawing of your data images. Be careful with sending messages from this callback to your application. lpfnEndFrameCallback and lpfnEndAcqCallback are called from a separate thread which is dissimilar to the applications main thread. That should cause problems if you send messages to your main thread via SendMessage. If this causes problems use PostMessage instead. If this parameter is set to NULL it is ignored.

lpfnEndAcqCallback

    Defines a function pointer that is called after the XISL did the sorting. The prototype for the function is given by:

void CALLBACK OnEndAcqCallback(HACQDESC hAcqDesc);

    In this routine you can perform any clean up at acquisition end. If this parameter is set to NULL, it is ignored.

7.2.8   Acquisition_Acquire_Image

This function acquires *dwFrames* frames and performs offset, gain and pixel corrections automatically. The routine returns immediately. If you want to be informed about frame end or acquisition end, then define in Acquisition_Init the suitable Callback functions and post from there a corresponding message to your application. The correction algorithms are described in more detail on page 14.

UINT WINAPI Acquisition_Acquire_Image(
      HACQDESC hAcqDesc,
      UINT dwFrames,
      UINT dwSkipFrames,
      UINT dwOpt,
      unsigned short *pwOffsetData,
      DWORD *pdwGainData,
      DWORD *pdwPixelData
      );

hAcqDesc
      Pointer to acquisition descriptor structure

dwFrames
      Number of frames to acquire is one of the sequence options is set for *dwOpt*. If the continuous option is set this value gives the number of frames in a ring buffer that is used for continuous data acquisition.

dwSkipFrames
      Number of frames to skip before a frames is copied into the acquisition buffer.

dwOpt
      Options for sequence acquisition: Valid values are

HIS_SEQ_TWO_BUFFERS	0x1	Storage of the sequence into two buffers. Secure image acquisition by separated data transfer and later performed image correction.
HIS_SEQ_ONE_BUFFER	0x2	Storage of the sequence into one buffer. Direct acquisition and linked correction into one buffer.
HIS_SEQ_AVERAGE	0x4	All acquired single images are directly added into one buffer and after acquisition divided by the number of frames, including linked correction files.
HIS_SEQ_DEST_ONE_FRAME	0x8	Sequence of frames using the same image buffer
HIS_SEQ_COLLATE	0x10	Skip frames after acquiring frames in a ring buffer
HIS_SEQ_CONTINUOUS	0x100	Continuous acquisition Frames are continuously acquired into a ring buffer of *dwFrames*

Table 52: Supported Acquisition Sequence Options

pwOffsetData
      Pointer that contains offset data. (see Acquisition_Acquire_OffsetImage). The Offset must be actual. It is recommended to acquire them shortly before calling **Acquisition_Acquire**. If you don't want to perform an offset correction set this parameter to NULL.

pdwGainData
      Pointer that contains gain data. (see Acquisition_Acquire_GainImage). If you don't want to perform a gain correction set this parameter to NULL.

pdwPixelData
      Pointer to a buffer that contains pixel correction data.pdwPixelData points to a linear array of data. Its size is given through ((number of wrong pixels) * 10 + 1) * sizeof(int). The first entry in a group of nine contains the offset of the pixel from the base pointer of the data array. The other eight entries equal the offset of the correction pixels from the base pointer. If you want to use less than eight pixels for correction, then set the remaining entries to -1. The value of the pixel is replaced by the mean value of the correction pixels. If you don't want to perform a pixel correction set this parameter to

NULL. An easier way to create a pixel map is the use of the XISL function
Acquisition_CreatePixelMap.

Return values

If the function is successful it returns zero, otherwise an error code. To get extended
information call Acquisition_GetErrorCode.

7.2.9   Acquisition_Acquire_Image_PreloadCorr

The function provides the same functionality as Acquisition_Acquire_Image(…) except loading
the image correction data. Use **Acquisition_SetCorrData_Ex** before to set the correction data.

UINT WINAPI RETURN Acquisition_Acquire_Image_PreloadCorr (
    HACQDESC hAcqDesc,
    UINT nFrames,
    UINT dwSkipFrms,
    UINT dwOpt
    );

Please find the parameter description above. (Acquisition_Acquire_Image(..))

7.2.10   Acquisition_Abort

This routine aborts a currently running acquisition.

UINT WINAPI Acquisition_Abort(
    HACQDESC hAcqDesc
    );

hAcqDesc

Pointer to acquisition descriptor structure

Return values

If the function is successful it returns zero, otherwise an error code. To get extended
information call Acquisition_GetErrorCode.

7.2.11   Acquisition_AbortCurrentFrame

This routine aborts the transfer of the current frame from the detector to the frame grabber. The
detector will be reset then and a new frame transfer will be started immediately.
The detector should run in the same mode as the image correction files have been obtained.
Therefore it is not recommended to use this function during a measurement.

UINT WINAPI Acquisition_AbortCurrentFrame(
    HACQDESC hAcqDesc
    );

hAcqDesc

Pointer to acquisition descriptor structure

Return values

If the function is successful it returns zero, otherwise an error code. To get extended
information call Acquisition_GetErrorCode.

7.2.12 Acquisition_SetCameraMode

This function sets the acquisition mode of the detector. Currently eight fixed frame times of the detector are provided.

UINT WINAPI Acquisition_SetCameraMode(
  HACQDESC hAcqDesc,
  UINT dwMode
  );

hAcqDesc
  Pointer to acquisition descriptor structure

dwMode
  Must be a number between 0 and 7. The corresponding frame time depends on the used
  PROM. (see frame times).

Return values
  If the function is successful it returns zero, otherwise an error code. To get extended
  information call Acquisition_GetErrorCode.

7.2.13 Acquisition_SetCorrData

This function switches the correction buffers during a running acquisition. It is also possible to switch off corrections by setting the pOffsetData, pGainData and pPixelCorrList to NULL.

UINT WINAPI Acquisition_SetCorrData(
  HACQDESC hAcqDesc,
  WORD *pOffsetData,
  DWORD *pGainData,
  DWORD *pPixelCorrList
  );

hAcqDesc
  Pointer to acquisition descriptor structure

pOffsetData
  Point to offset data (see Acquisition_Acquire_OffsetImage).

pGainData
  Pointer that contains gain data (see Acquisition_Acquire_GainImage).

pPixelCorrList
  Pointer to a pixel correction list (see Acquisition_DoPixelCorrection).

Return values
  If the function is successful it returns zero, otherwise an error code. To get extended
  information call Acquisition_GetErrorCode.

7.2.14   Acquisition_Acquire_OffsetImage

This function acquires *nFrames*, adds them in a 32 bit buffer and after acquisition the data values
are divided by *nFrames* (averaging). The last acquired data at frame end time are available via
pOffsetData. At the end of the acquisition time the averaged data are also accessible via *pOffsetData*.
The routine returns immediately. If you want to be informed about frame end or acquisition end
then define in Acquisition_Init the suitable Callback functions and post from there a
corresponding message to your application.

UINT WINAPI Acquisition_Acquire_OffsetImage(
    HACQDESC hAcqDesc,
    unsigned short *pOffsetData,
    UINT nRows,
    UINT nCols,
    UINT nFrames
    );

hAcqDesc
    Pointer to acquisition descriptor structure.
pOffsetData
    Pointer to a acquisition buffer for offset data.
nFrames
    Number of frames to acquire.
nRows, nCols
    Number of rows and columns of the offset data buffer. If the values are not suitable to the
    current connected sensor the function return with an error.
Return values
    If the function is successful it returns zero, otherwise an error code. To get extended
    information call Acquisition_GetErrorCode.

7.2.15   Acquisition_Acquire_OffsetImage_PreloadCorr

The function provides the same functionality as Acquisition_Acquire_Offset_Image(…) except
loading the image correction data. Use **Acquisition_SetCorrData_Ex** before to all correction
data to NULL.

UINT WINAPI Acquisition_Acquire_OffsetImage_PreloadCorr (
    HACQDESC hAcqDesc,
    unsigned short *pwOffsetData,
    UINT nRows,
    UINT nColumns,
    UINT nFrames,
    UINT dwOpt);
    Please find the parameter description above. (Acquisition_Acquire_OffsetImage(..))
    UINT dwOpt must be 0
    );

7.2.16　　　Acquisition_Acquire_GainImage

This function acquires *nFrames* which are all offset corrected by data stored in *pOffsetData*. After that the gain data are added in a 32 bit buffer and after acquisition the data values are divided by *nFrames* (averaging). After averaging the data are further processed for subsequent gain correction of image data. The last acquired data at frame end time are available via *pGainData*. At the end of the acquisition time the gain data are also accessible via *pGainData*.

The offset data are necessary to derive a valid gain image.

The routine returns immediately. If you want to be informed about frame end or acquisition end then define in Acquisition_Init the suitable Callback functions and post from there a corresponding message to your application.

UINT WINAPI Acquisition_Acquire_GainImage(
　　　HACQDESC hAcqDesc,
　　　WORD *pOffsetData,
　　　DWORD *pGainData,
　　　UINT nRows,
　　　UINT nCols,
　　　UINT nFrames
　　　);

hAcqDesc
　　　Pointer to acquisition descriptor structure.
pOffsetData
　　　Pointer that contains offset data. (see Acquisition_Acquire_OffsetImage). It is
　　　recommended to acquire the Offset shortly before calling
　　　Acquisition_Acquire_GainImage.
pGainData
　　　Pointer to buffer that receives the gain data.
nFrames
　　　Number of frames to acquire.
nRows, nCols
　　　Number of rows and columns of the offset data buffer. If the values are not suitable to the
　　　current connected sensor the function return with an error.
Return values
　　　If the function is successful it returns zero, otherwise an error code. To get extended
　　　information call Acquisition_GetErrorCode.

7.2.17　　　Acquisition_Acquire_GainImage_PreloadCorr

The function provides the same functionality as Acquisition_Acquire_Gain_Image(…) except loading the image correction data. Use **Acquisition_SetCorrData_Ex** before to set Offset Correction data only.

UINT WINAPI Acquisition_Acquire_GainImage_PreloadCorr(
　　　HACQDESC hAcqDesc,
　　　DWORD *pGainData,
　　　UINT nRows,
　　　UINT nCols,
　　　UINT nFrames
　　　);

Please find the parameter description above. (Acquisition_Acquire_GainImage(..))

7.2.18   Acquisition_CreatePixelMap

This function specifies the size of or creates a pixel correction map (depending on pCorrList) that one can use in Acquisition_Acquire_Image or Acquisition_DoPixelCorrection.

UINT WINAPI Acquisition_CreatePixelMap(
WORD *pData,
int nDataRows,
int nDataColumns,
int *pCorrList,
int *pnCorrListSize
);

pData

Pointer to a data source buffer. Defective pixels are marked by setting their value to −1 (0xFFFF). All other pixel values are recognized as good ones.

nDataRows

Number of rows of the data source.

nDataColumns

Number of columns of the data source.

pCorrList

Pointer to an array (pixel map buffer) that receives the pixel correction data. If pCorrlist is set to NULL then only the required size of the pixel map buffer is returned in pnCorrListSize.

pnCorrListSize

Pointer to an integer that receives the required size of the pixel map buffer if pCorrList is set to NULL otherwise it contains the size of the pixel buffer at function call time.

7.2.19   Acquisition_DoOffsetCorrection

This function performs an offset correction for the data defined in Acquisition_DefineDestBuffers. A suitable place to call this function is the end of frame callback function defined by Acquisition_Init. The correction algorithms are described in more detail on page 14.

UINT WINAPI Acquisition_DoOffsetCorrection(
WORD *pSource,
WORD *pDest,
WORD *pOffsetData,
int nCount
);

pSource

Pointer to data source buffer.

pDest

Pointer to data destination buffer. This parameter can be equal to pSource.

pOffsetData

Pointer that contains offset data. (see Acquisition_Acquire_OffsetImage). These data must be actual. It is recommended to acquire them shortly before calling Acquisition_DoOffsetCorrection.

nCount

Number of pixels to correct.

Return values

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.21 Acquisition_DoOffsetGainCorrection

This function performs an offset and a gain correction for the data defined in Acquisition_DefineDestBuffers at once. A suitable place to call this function is the end of frame callback function defined by Acquisition_Init. The correction algorithms are described in more detail on page 14.

UINT WINAPI Acquisition_DoOffsetGainCorrection(
  WORD *pSource,
  WORD *pDest,
  WORD *pOffsetData,
  DWORD *pGainData,
  int nCount
  );

pSource
  Pointer to data source buffer.
pDest
  Pointer to data destination buffer. This parameter can be equal to pSource.
pOffsetData
  Point to offset data. (see Acquisition_Acquire_OffsetImage).
pGainData
  Pointer that contains gain data. (see Acquisition_Acquire_GainImage).
nCount
  Number of data entries to correct.
Return values
  If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.22   Acquisition_DoPixelCorrection

This function performs a pixel correction for the data defined in Acquisition_DefineDestBuffers. A suitable place to call this function is the end of frame callback function defined by Acquisition_Init. The correction algorithms are described in more detail on page 14.

UINT WINAPI Acquisition_DoPixelCorrection(
    WORD *pData,
    int *pCorrList
    );

pData
    Pointer to data.
pCorrList
    Pointer that contains correction data. pCorrList points to a linear array of data. Its size is given through ((number of pixels) * 9 + 1). The first entry in a group of nine contains the offset of the pixel from the base pointer of the data array. The other eight entries equal the offset of the correction pixels from the base pointer. If you want to use less than eight pixels for correction, then set the remaining entries to -1. The value of the pixel is replaced by the mean value of the correction pixels. The end of the pixel correction list is indicated by a value of -1 as the last entry in the pixel map.
Return values
    If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.23   Acquisition_IsAcquiringData

This function tests if XISL is about to acquire data.
UINT WINAPI Acquisition_IsAcquiringData(
    HACQDESC hAcqDesc
    );
hAcqDesc
    Pointer to acquisition descriptor structure.
Return values
    If an acquisition is running a one is returned, otherwise zero.

7.2.24   Acquisition_GetErrorCode

The function returns extended information if an error occurred during a XISL function call.
UINT WINAPI Acquisition_GetErrorCode(
    HACQDESC hAcqDesc,
    DWORD *dwHISError**,**
    DWORD *dwBoardError
    );

hAcqDesc
    Pointer to acquisition descriptor structure.
dwHISError
    Retrieves an error code regarding the XISL itself.
dwBoardError
    Retrieves an error code regarding the acquisition board. Please consult the corresponding documentation of your data acquisition board.
Return values
    If the function is successful it returns zero otherwise an error code.

7.2.25   Acquisition_Close

Hardware and XISL are closed by this routine. The acquisition descriptor is no longer valid.

UINT WINAPI Acquisition_Close(
      HACQDESC hAcqDesc
      );

hAcqDesc
      Pointer to acquisition descriptor structure.

Return values
      If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.26   Acquisition_CloseAll

This function closes shuts down all frame grabbers and serial interfaces currently allocated by the XISL. All acquisition descriptor structures returned by other functions are invalid after this function call.

UINT WINAPI Acquisition_CloseAll();

Return values
      If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.27   Acquisition_SetReady

This function must be called when the application finished redrawing of the new acquired data. A good place to call this function is the end of frame callback function defined in Acquisition_Init or the message handler to for the redraw message after redrawing.

UINT WINAPI Acquisition_SetReady(
      HACQDESC hAcqDesc,
      BOOL bFlag
      );

hAcqDesc
      Pointer to acquisition descriptor structure.

bFlag
      Boolean value. Set to zero to signal XISL set redrawing isn't ready, otherwise set to one.

Return values
      If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.28   Acquisition_GetReady

This function checks whether the passed Acquisition Descriptor is valid.

UINT WINAPI Acquisition_GetReady(
      HACQDESC hAcqDesc
      );

hAcqDesc
      Pointer to acquisition descriptor structure.

Return values
      If the Acquisition Descriptor is valid, the function returns zero; otherwise it returns an error code. To get extended information call Acquisition_GetErrorCode.

7.2.29   Acquisition_GetConfiguration

This function retrieves all important acquisition parameters, that can be set by Acquisition_Init or that are set by the self configuration mechanisms of the XISL.

UINT WINAPI Acquisition_GetConfiguration(
> **HACQDESC** hAcqDesc,
> **UINT** *dwFrames,
> **UINT** *dwRows,
> **UINT** *dwColumns,
> **UINT** *dwDataType,
> **UINT** *dwSortFlags,
> **BOOL** *bIRQEnabled,
> **DWORD** *dwAcqType,
> **DWORD** *dwSystemID,
> **DWORD** *dwSyncMode,
> **DWORD** *dwHwAccess
> **);**

hAcqDesc
> Pointer to acquisition descriptor structure.

dwFrames
> Number of frames of acquisition buffer.

dwRows, dwColumns
> Number of rows and columns of the sensor.

dwDataType
> Type of data of acquisition buffer (should always be two = unsigned short).

dwSortFlags
> Type of sorting. This value depends on detector and used sensor (see sorting schemes).

bIRQEnabled
> Retrieves a flag that indicates if interrupts are enabled (see Acquisition_Init,. hardware interrupts) or if the hardware is running in polling mode. In interrupt mode this parameter is equal to one and zero in the other case.

dwAcqType
> Only for internal use.

dwSystemID
> PROM identification number of the used detector. This number is only important if the detector operates objectionably and you need any support from PerkinElmer.

dwSyncMode
> This parameter can receive the following values:

HIS_SYNCMODE_FREE_RUNNING	The sensor is operating in free running mode.
HIS_SYNCMODE_EXTERNAL_TRIGGER	The sensor is operating in triggered mode. Frames are only send if an external trigger signal is applied to the detector.
HIS_SYNCMODE_INTERNAL_TIMER	The synchronization signal can be generated by the internal timer of the frame grabber (see Acquisition_SetTimerSync)
HIS_SYNCMODE_SOFT_TRIGGER	The synchronization signal can be generated by software (see Acquisition_SetFrameSync).
HIS_SYNCMODE_AUTO_TRIGGER[5]	The detector triggers itself by x-ray exposure.

Table 53: Description of the SyncMode Options

dwHwAccess
> This parameter receives the hardware access parameter, that is programmed into the detector.
> If you have to use this parameter (that depends from your contract with PerkinElmer) please contact PerkinElmer for the possible values.

Return values
> If the function is successful it returns zero otherwise an error code. To get extended information call Acquisition_GetErrorCode.

---

[5] Only for XRpad detector series available

7.2.30   Acquisition_GetIntTimes

This function retrieves the current integration times.

UINT WINAPI Acquisition_GetIntTimes(
  HACQDESC hAcqDesc,
  double *pdblIntTime,
  int *nIntTimes
  );

hAcqDesc
  Pointer to acquisition descriptor structure.
pdblIntTime
  Pointer to an array of 8 byte floating point numbers. This array must contain at least 8
  entries.
nIntTimes
  This parameter contains the number of maximum entries in the array of 8 byte floating
  point numbers pointed to by *pdblIntTime*. After return of the function this variable
  provides the number of available integration times.
Return values
  If the function is successful it returns zero, otherwise an error code. To get extended
  information call Acquisition_GetErrorCode.

Example:
```
double dblIntTimes[8];
int nIntTimes = 8;
if (Acquisition_GetIntTimes(hAcqDesc, dblIntTimes, &nIntTimes)!=HIS_ALL_OK)
{
  //error handling
}
printf("Number of available integration times: %d\n", nIntTimes);
for (int i=0; i<nIntTimes; i++)
{
        printf("%d: %f\n",i, dblIntTimes[i]);
}
```

7.2.31 Acquisition_SetAcqData

This routine sets a 32 bit integer that can be received with Acquisition_GetAcqData. These two functions are useful to avoid global variables to put through parameters to the end of frame and end of acquisition callback functions set by Acquisition_Init.

You can use this function to pass parameters to your callback routines. Thereby it is possible to avoid global variables. The recommended way of use is to define a data structure/value/class and to pass the address of an instance of it by Acquisition_SetAcqData. This has the advantage that a variety of parameters, which are defined within the structure/value/class, are accessible within the callback functions.

The functions Acquisition_SetAcqData and Acquisition_GetAcqData differ in the 32 and the 64 bit library:

32bit:
UINT WINAPI Acquisition_SetAcqData(
      HACQDESC hAcqDesc
      DWORD dwData
      );

64bit:
UINT WINAPI Acquisition_SetAcqData(
      HACQDESC hAcqDesc
      void* AcqData
      );

hAcqDesc
      Pointer to acquisition descriptor structure.

32bit:
dwData
      Data to be accessible within callback functions. dwData can represent a single value as well as an address. In that case a structure/value/class has to be defined with the required parameters and cast *dwData* to the pointer.
      The value/ address can be retrieved in the EndFrameCallback/ EndAcqCallback by using Acquisition_GetAcqData(..).

64bit:
AcqData
      Data to be accessable within callback functions: Pass a pointer to a user defined structure/value/class which can be retrieved in the EndFrameCallback/ EndAcqCallback by using Acquisition_GetAcqData(..)

Return values
      If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.32   Acquisition_GetAcqData

This routine returns a 32 bit integer that can be set by Acquisition_SetAcqData. These two functions are useful to avoid global variables to put through parameters to the end of frame and end of acquisition callback functions set by Acquisition_Init.

The functions Acquisition_SetAcqData and Acquisition_GetAcqData differ in the 32 and the 64 bit library:

32bit:

UINT WINAPI Acquisition_GetAcqData(
    HACQDESC hAcqDesc
    DWORD *dwData
    );

64bit:

UINT WINAPI Acquisition_SetAcqData(
    HACQDESC hAcqDesc
    void **VoidAcqData
    );

hAcqDesc
    Pointer to acquisition descriptor structure.

32bit:

dwData
    Data to be received. To put through more than one parameters define a structure with the required parameters and cast the pointer to dwData.

64bit:

VoidAcqData
    Pointer to a Pointer to a user defined structure/value/class here which can be retrieved in the EndFrameCallback/ EndAcqCallback.

Return values
    If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.33   Acquisition_GetActFrame

This function retrieves the current acquisition frames.

UINT WINAPI Acquisition_GetActFrame(
    HACQDESC hAcqDesc,
    DWORD *dwActAcqFrame,
    DWORD *dwActBuffFrame
    );

hAcqDesc
    Pointer to acquisition descriptor structure.

dwActAcqFrame
    Index of the latest acquired frame within the internal acquisition buffer. The buffer itself is part of the frame grabber driver. Its data cannot accessible externally. However, the index can be used to verify the consistency of image sequences; it runs repeatently from 1 to 8.

dwActBuffFrame
    Index of the latest acquired frame within the user buffer. For that, size and location of the user buffer has to be defined by *Acquisition_DefineDestBuffers*.

Return values
    If the function is successful it returns zero otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.34 Acquisition_SetFrameSyncMode

This function sets the synchronization mode of the frame grabber. (see 6.1Readout schema)

UINT WINAPI Acquisition_SetFrameSyncMode(
    HACQDESC hAcqDesc,
    DWORD dwMode
    );

hAcqDesc

    Pointer to acquisition descriptor structure.

dwMode

This parameter can have the values:

HIS_SYNCMODE_FREE_RUNNING
HIS_SYNCMODE_EXTERNAL_TRIGGER
HIS_SYNCMODE_INTERNAL_TIMER
HIS_SYNCMODE_SOFT_TRIGGER
HIS_SYNCMODE_AUTO_TRIGGER[6]

Table 54: Trigger option Flags

Return values

    If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.35 Acquisition_SetFrameSync

This function supplies a synchronization signal (soft-trigger) to the detector every time this function is called.

UINT WINAPI Acquisition_SetFrameSync(
    HACQDESC hAcqDesc
    );

hAcqDesc

    Pointer to acquisition descriptor structure.
    Before calling this function you have to set the frame grabber to a suitable synchronization mode by a call of Acquisition_SetFrameSyncMode.

Return values

    If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.36 Acquisition_SetTimerSync

This function supplies a synchronization signal to the detector every time dwCycleTime is over.

UINT WINAPI Acquisition_SetTimerSync(
    HACQDESC hAcqDesc,
    DWORD *dwCycleTime
    );

hAcqDesc

    Pointer to acquisition descriptor structure.

dwCycleTime

    Pointer to a 32 bit integer that provides the required cycle time in μs. After returning of the function this parameter contains the realized cycle time.
    Before calling this function you have to set the frame grabber to a suitable synchronization mode by a call of Acquisition_SetFrameSyncMode.
    Some frame grabbers can realize synchronization cycles only in discreet steps. That's why the function returns the realized cycle time in *dwCycleTime.*

Return values

---

[6] Only for XRpad detector series available

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.37 Acquisition_SetFrameSyncTimeMode

This function sets the synchronization mode of the detector to triggered mode and sets the delay time for the "Data Delivered on Demand with[out] clearance scan " triggered mode with defined integration time.

UINT WINAPI Acquisition_SetFrameSyncTimeMode(
 HACQDESC hAcqDesc**,**
 unsigned int uiMode**,**
 unsigned int dwDelayTime
 );

hAcqDesc
 Pointer to acquisition descriptor structure.
uiMode
 Trigger Mode must be 0 .. 7
dwDelayTime
 Additional delay time in milliseconds (can be 0-ms up to (5000 ms-intTime)
Return values
 If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

Example:

```
// set special triggermode to 'Data Delivered on demand without clearance scan'
Acquisition_SetCameraTriggerMode (hAcqDesc,1);


// set Framegrabber to Soft_Trigger-Mode
Acquisition_SetFrameSyncMode(hAcqDesc,HIS_SYNCMODE_SOFT_TRIGGER);

//set detector to timing 0 delay 1sec
Acquisition_SetFrameSyncTimeMode(hAcqDesc,0,1000);

hevEndAcq = CreateEvent(NULL, FALSE, FALSE, NULL);
if ((nRet=Acquisition_Acquire_Image(hAcqDesc, 1, 0, HIS_SEQ_ONE_BUFFER,
NULL, NULL, NULL))!=HIS_ALL_OK)
{
                //error handling
}

//  start the readout
Acquisition_SetFrameSync(hAcqDesc);
```

7.2.38 CHwHeaderInfo

Structure that is used to retrieve the contents of detector's hardware header by Acquisition_GetHwHeader.

typedef struct
{
```
    DWORD           dwPROMID;
    DWORD           dwHeaderID;
    BOOL            bAddRow;
    BOOL            bPwrSave;
    DWORD           dwNrRows;
    DWORD           dwNrColumns;
    DWORD           dwZoomULRow;
    DWORD           dwZoomULColumn;
    DWORD           dwZoomBRRow;
    DWORD           dwZoomBRColumn;
    DWORD           dwFrmNrRows;
    DWORD           dwFrmRowType;
    DWORD           dwFrmFillRowIntervalls;
    DWORD           dwNrOfFillingRows;
    DWORD           dwDataType;
    DWORD           dwDataSorting;
    DWORD           dwTiming;
    DWORD           dwAcqMode;
    DWORD           dwGain;
    DWORD           dwOffset;
    BOOL            bSyncMode;
    DWORD           dwBias;
    DWORD           dwLeakRows;
} CHwHeaderInfo;
```

DWORD	dwPROMID;	identifies the detector's PROM set
DWORD	dwHeaderID;	identification number of this header type.
BOOL	bAddRow;	indicates if an additional row is transferred
BOOL	bPwrSave;	indicates if detector is in power safe mode
DWORD	dwNrRows;	number of sensor rows
DWORD	dwNrColumns;	number of sensor columns
DWORD	dwZoomULRow;	row of the upper left edge of zoom region
DWORD	dwZoomULColumn;	column of the upper left edge of zoom region
DWORD	dwZoomBRRow;	row of bottom right edge of zoom region
DWORD	dwZoomBRColumn;	column of bottom right edge of zoom region
DWORD	dwFrmNrRows;	Number of rows that are used to synthesize the frame scheme of the detector. It results from the number of sensor rows plus the number of rows in which the sensor only integrates charge but doesn't transfer data to the frame grabber plus the number of filling rows.
DWORD	dwFrmRowType;	see Row Types
DWORD	dwFrmFillRowIntervalls;	Intervals of 10 nanoseconds to synthesize a frame (see description of hardware header (Header ID <11 otherwise used for int time detection in some header versions)
DWORD	dwNrOfFillingRows	Number of rows of the above mentioned row time.
DWORD	dwDataType;	used for int time detection in some header versions
DWORD	dwDataSorting;	see sorting
DWORD	dwTiming;	selected integration time
DWORD	dwAcqMode;	fixed mode (0), sync mode (1) with fixed frame regime (Header ID <11)
DWORD	dwGain;	1 for Header ID < 11 otherwise Gain mode
DWORD	dwOffset;	
BOOL	bSyncMode	1 if the detector operates in triggered mode else 0.
DWORD	dwBias;	10 V * SensorBias / 255 (Header ID <11)
DWORD	DwLeakRows	Number of rows without driven gates

Table 55: Description of the ChwHeaderInfo structure

BYTE is an unsigned character, USHORT an unsigned 16 bit integer. If HeaderID is zero the whole hardware header is invalid.

For Header-ID 10 the sensor frame time Tint can be derived by the following formula:

Tint =

(LSBNumberRows+MSBNumberRows*65536)*RowTime+CycleTime*NumberOfEmptyCycles

To get the value for RowTime see Row Types. CycleTime equals 15.625 nanoseconds.

During acquisition time a direct access to the hardware header is possible by the XISL function Acquisition_GetHwHeader(..).

For Header-ID 11 and 12 the frame time is

Tint = dwFrmNrRows * dwFrmFillRowIntervalls/64)

For Header-ID 13 the frame time is

Tint = dwFrmNrRows * (1/ dwDataType) * 32.0 * (dwFrmFillRowIntervalls/64))

For Header-ID 14 (extended header values are used) the frame time is

Tint = dwFrmNrRows * (1 / (wClock /64)) * 32.0 * (wRowTime /64);

7.2.39 Acquisition_GetHwHeaderInfo

This function returns the contents of the detector's hardware header in a CHwHeaderInfo structure.

UINT WINAPI Acquisition_GetHwHeaderInfo(
      HACQDESC hAcqDesc,
      CHwHeaderInfo *pInfo
      );

hAcqDesc
      Pointer to acquisition descriptor structure.

pInfo
      Structure of type CHwHeaderInfo that contains the contents of the detector's hardware header necessary for self configuration features.

Return values
      If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.40 Acquisition_GetWinHandle

This function retrieves the current acquisition window handle.

UINT WINAPI Acquisition_GetWinHandle(
      HACQDESC hAcqDesc,
      HWND *hWnd
      );

hAcqDesc
      Pointer to acquisition descriptor structure.

hWnd
      Retrieves the window handle defined in Acquisition_Init.

Return values
      If the function is successful it returns zero otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.41 Acquisition_CreateGainMap

This function creates a list of median values to be used with the function
Acquisition_Acquire_Image_Ex. One value for each image in the pGainData-sequence

UINT WINAPI Acquisition_CreateGainMap(

> **WORD** *pGainData,
> **WORD** *pGainAVG,
> **int** nCount,
> **int** nFrame
> **);**

pGainData

> pointer to a sequence of offset corrected data. The images in the array must be sorted by
> median.

pGainAVG

> pointer to a *nFrame* sized array of type word

nCount

> Number of pixels to per image.

NFrame

> Number of Frames in pGainData

Return values

> If the function is successful it returns true otherwise false.

7.2.42 Acquisition_Acquire_Image_Ex

This function acquires *dwFrames* frames and performs offset, gain/gain-sequence and pixel
corrections automatically. The routine returns immediately. If you want to be informed about
frame end or acquisition end, then define in Acquisition_Init the suitable Callback functions and
post from there a corresponding message to your application. The correction algorithms are
described in more detail on page 14.

UINT WINAPI Acquisition_Acquire_Image_Ex(

> | **HACQDESC** | hAcqDesc, |
> |---|---|
> | **UINT** | dwFrames, |
> | **UINT** | dwSkipFrms, |
> | **UINT** | dwOpt, |
> | **unsigned short** | *pwOffsetData, |
> | **UINT** | dwGainFrames, |
> | **unsigned short** | *pwGainData, |
> | **unsigned short** | *pwGainAvgData, |
> | **DWORD** | *pdwGainData, |
> | **DWORD** | *pdwPxlCorrList |

> **);**

hAcqDesc

> Pointer to acquisition descriptor structure

dwFrames

> Number of frames to acquire is one of the sequence options is set for *dwOpt*. If the
> continuous option is set this value gives the number of frames in a ring buffer that is used
> for continuous data acquisition.

dwSkipFrames

> Number of frames to skip before a frames is copied into the acquisition buffer.

dwOpt

> Options for sequence acquisition: Valid values are

HIS_SEQ_TWO_BUFFERS	0x1	Storage of the sequence into two buffers. Secure image acquisition by separated data transfer and later performed image correction.
HIS_SEQ_ONE_BUFFER	0x2	Storage of the sequence into one buffer.

		Direct acquisition and linked correction into one buffer.
HIS_SEQ_AVERAGE	0x4	All acquired single images are directly added into one buffer and after acquisition divided by the number of frames, including linked correction files.
HIS_SEQ_DEST_ONE_FRAME	0x8	Sequence of frames using the same image buffer
HIS_SEQ_COLLATE	0x10	Skip frames after acquiring frames in a ring buffer
HIS_SEQ_CONTINUOUS	0x100	Continuous acquisition Frames are continuously acquired into a ring buffer of *dwFrames*

Table 56 : Description of the SyncMode Options

pwOffsetData

Pointer that contains offset data. (see Acquisition_Acquire_OffsetImage). The Offset must be actual. It is recommended to acquire them shortly before calling Acquisition_Acquire. If you don't want to perform an offset correction set this parameter to NULL.

dwGainFrames,

number of frames in pwGainData

pwGainData

pointer to a sequence of offset corrected data. The images in the array must be sorted by median.

pwGainAvgData

pointer to the median-list created by Acquisition_CreateGainMap

pdwGainData

Pointer that contains gain data. (see Acquisition_Acquire_GainImage). If you don't want to perform a gain correction set this parameter to NULL.

pdwPxlCorrList

Pointer to a buffer that contains pixel correction data.pdwPixelData points to a linear array of data. Its size is given through ((number of wrong pixels) * 10 + 1) * sizeof(int). The first entry in a group of nine contains the offset of the pixel from the base pointer of the data array. The other eight entries equal the offset of the correction pixels from the base pointer. If you want to use less than eight pixels for correction, then set the remaining entries to -1. The value of the pixel is replaced by the mean value of the correction pixels. If you don't want to perform a pixel correction set this parameter to NULL. An easier way to create a pixel map is the use of the XISL function Acquisition_CreatePixelMap.

Return values

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.43 Acquisition_SetCorrData_Ex

This function switches the correction buffers during a running acquisition. It is also possible to switch off corrections by setting the pOffsetData, pGainData *pwGainData*, *pwGainAvgData*, *nGainFrames*, and pPixelCorrList to NULL.

UINT WINAPI Acquisition_SetCorrData_Ex(
    HACQDESC hAcqDesc,
    unsigned short *pwOffsetData,
    unsigned short *pwGainData,
    unsigned short *pwGainAvgData,
    UINT nGainFrames,
    DWORD *pdwGainData,
    DWORD *pdwPxlCorrList
    );

hAcqDesc

Pointer to acquisition descriptor structure

pwOffsetData

>   Pointer to offset data (see Acquisition_Acquire_OffsetImage).

pwGainData

>   Pointer to a sequence of offset corrected data. The images in the array must be sorted by
>   median.

pwGainAvgData

>   Pointer to the median-list created by **Acquisition_CreateGainMap**

nGainFrames

>   number of frames in pwGainData

pdwGainData

>   Pointer that contains gain data (see Acquisition_Acquire_GainImage).

pdwPxlCorrList)

>   Pointer to a pixel correction list (see Acquisition_DoPixelCorrection)

Return values

>   If the function is successful it returns zero, otherwise an error code. To get extended
>   information call Acquisition_GetErrorCode.

7.2.44   Acquisition_GetCorrData_Ex

This function retrieves the current correction data during a running acquisition.

UINT WINAPI Acquisition_GetCorrData_Ex(

HACQDESC	hAcqDesc,
unsigned short	**ppwOffsetData,
unsigned short	**ppwGainData,
unsigned short	**ppwGainAvgData,
UINT	**nGainFrames,
DWORD	**ppdwGainData,
DWORD	**ppdwPxlCorrList

);

hAcqDesc

>   Pointer to acquisition descriptor structure

ppwOffsetData

>   Pointer to offset data (see Acquisition_Acquire_OffsetImage).

ppwGainData

>   Pointer to a sequence of offset corrected data. The images in the array must be sorted by
>   median.

ppwGainAvgData

>   Pointer to the median-list created by **Acquisition_CreateGainMap**

nGainFrames

>   UINT *pointer retrieving the number of frames in pwGainData

ppdwGainData

>   Pointer that contains gain data (see Acquisition_Acquire_GainImage).

ppdwPxlCorrList)

>   Pointer to a pixel correction list (see Acquisition_DoPixelCorrection)

Return values

>   If the function is successful it returns zero, otherwise an error code. To get extended
>   information call Acquisition_GetErrorCode.

7.2.45   Acquisition_DoOffsetGainCorrection_Ex

This function performs an offset and a gain correction for the data defined in
Acquisition_DefineDestBuffers at once. A suitable place to call this function is the end of frame
callback function defined by Acquisition_Init. The correction algorithms are described in more
detail on page 14.

UINT WINAPI Acquisition_DoOffsetGainCorrection_Ex(

       WORD *pSource,

       WORD *pDest,

       WORD *pOffsetData,

       WORD *pGainData,

       WORD *pGainAVG,

       int   nCount,

       int   nFrame

       );

pSource

       Pointer to data source buffer

pDest

       Pointer to data destination buffer. This parameter can be equal to pSource.

pOffsetData

       Pointer to offset data. (see Acquisition_Acquire_OffsetImage).

pGainData

       *P*ointer to a sequence of offset corrected data. The images in the array must be sorted by median.

pGainAVG

       Pointer to the median-list created by **Acquisition_CreateGainMap**

nCount

       Number of data entries to correct. (rows*cols)

nFrame

       number of frames in pGainData

Return values

       If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.46   Acquisition_SetCameraGain

This function can be used to set the gain factor of the detector.

Acquisition_SetCameraGain(

       HACQDESC hAcqDesc,

       WORD wMode

       );

hAcqDesc

       Pointer to acquisition descriptor structure

wmode

       Gain factor to set.

       For the XRpad please refer to the XRpad reference manual.

       For the AM-Type the values of all capacities are added. All bitwise combinations are valid. For example: 3 => 1.3pF.

       For the xN/xO/xP-Type the Value in the table is set when the detector provides the functionality (refer to detector specification).

	Detector 16x0 AM
0	0.1pF (always on)
1	0.3pF
2	0.9pF
4	4.7pF
8	10pF
	Detector xN/xO/xP
0	0.25pF
1	0.5pF
2	1 pF
3	2 pF
4	4 pF
5	8 pF

Table 57: Detector gain values

Return values

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.47 Acquisition_ Acquire_GainImage_EX_ROI

This function is similar to the Acquisiton_Acquire_GainImage(..)-function.
The function provides the possibility to use a well-defined region of interest for the median determination. The median is used to calculate the gain of single pixels. (see Mathematical description of corrections) The function should be used to acquire the gain image when not the whole panel is illuminated.

UINT WINAPI Acquisition_Acquire_GainImage_EX_ROI (

HACQDESC	hAcqDesc,
WORD	*pOffsetData,
DWORD	*pGainData,
UINT	nRows,
UINT	nCols,
UINT	nFrames,
UINT	dwOpt,
UINT	uiULX, **UINT** uiULY, **UINT** uiBRX, **UINT** uiBRY,
UINT	uiMode
);	

hAcqDesc

Pointer to acquisition descriptor structure.

pOffsetData

Pointer that contains offset data. (see Acquisition_Acquire_OffsetImage). It is recommended to acquire the Offset shortly before calling **Acquisition_Acquire_GainImage**.

pGainData

Pointer to buffer that receives the gain data.

nFrames

Number of frames to acquire.

nRows, nCols

Number of rows and columns of the offset data buffer. If the values are not suitable to the current connected sensor the function return with an error.

dwOpt

must be 0

uiULX, uiULY, uiBRX, uiBRY,

UpperLeftX, UpperLeftY ,BottomRightX, BottomRightY

define a rect which is used to calculate the Median for the pixel-gain calculation.

uiMode

0 - normal Gain whole image used for median determination.

1 - Median for pixel-gain calculation from ROI (defined by uiULX…)

2 - Median for pixel-gain calculation from ROI each pixel-gain outside ROI
will be set to 1

3 - Median for pixel-gain calculation from ROI each pixel-gain outside ROI
will be set to 0

Return values

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.48 Acquisition_Acquire_GainImage_Ex_ROI_PreloadCorr

The function provides the same functionality as Acquisition_Acquire_GainImage_EX_ROI(…) except loading the image correction data. Please use **Acquisition_SetCorrData_Ex(..)** to set the correction data before.

UINT WINAPI Acquisition_Acquire_GainImage_Ex_ROI_PreloadCorr(

HACQDESC	hAcqDesc,
DWORD	*pGainData,
UINT	nRows,
UINT	nCols,
UINT	nFrames,
UINT	dwOpt,
UINT	uiULX,**UINT** uiULY,**UINT** uiBRX,**UINT** uiBRY,
UINT	uiMode
);	

Please find the parameter description above.

7.2.49 CHwHeaderInfoEx

Structure that is used to retrieve the extended detector hardware header by Acquisition_GetHwHeaderINfoEx(..) or GetLatestFrameHeader(..). This function is only available for detectors with HeaderID 14 or above. (1621, 0820)

typedef struct{

    WORD wHeaderID**;**
    WORD wPROMID**;**
    WORD wResolutionX**;**
    WORD wResolutionY**;**
    WORD wNrRows**;**
    WORD wNrColumns**;**
    WORD wZoomULRow**;**
    WORD wZoomULColumn**;**
    WORD wZoomBRRow**;**
    WORD wZoomBRColumn**;**
    WORD wFrmNrRows**;**
    WORD wFrmRowType**;**
    WORD wRowTime**;**
    WORD wClock**;**
    WORD wDataSorting**;**
    WORD wTiming**;**
    WORD wGain**;**
    WORD wLeakRows**;**
    WORD wAccess**;**
    WORD wBias**;**
    WORD wUgComp**;**
    WORD wCameratype**;**
    WORD wFrameCnt**;**
    WORD wBinningMode**;**
    WORD wRealInttime_milliSec**;**
    WORD wRealInttime_microSec**;**
    WORD wStatus**;**
    WORD wCommand1;
    WORD wCommand2;
    WORD wCommand3;
    WORD wCommand4;

```
        WORD  wDummy;
}       CHwHeaderInfoEx;
```

Description of structure entries (All entries are 16 bit Values)

WORD	wHeaderID	identifies the used header version (>=14)	
WORD	wPROMID	identifies the detector's PROM set	
WORD	wResolutionX	detectors pixel resolution in x direktion	
WORD	wResolutionY	detectors pixel resolution in y direktion	
WORD	wNrRows	number of sensor rows	
WORD	wNrColumns	number of sensor columns	
WORD	wZoomULRow	row of the upper left edge of zoom region	
WORD	wZoomULColumn	column of the upper left edge of zoom region	
WORD	wZoomBRRow	row of bottom right edge of zoom region	
WORD	wZoomBRColumn	column of bottom right edge of zoom region	
WORD	wFrmNrRows	Number of rows that are used to synthesize the frame scheme of the detector. It results from the number of sensor rows plus the number of rows in which the sensor only integrates charge but doesn't transfer data to the frame grabber.	
WORD	wFrmRowType	Identifies the implemented Row scheme	
WORD	wRowTime	detector row time in 32MhZ Ticks (6bit shifted to the left)	
WORD	wClock	detector row time in Mhz (6bit shifted to the left)	
WORD	wDataSorting	see sorting	
WORD	wTiming;	selected integration time	
WORD	wGain;	Selected detector gain (see table 6.1.10)	
WORD	wLeakRows;	Number of rows without driven gates	
WORD	wAccess;	Access mode	
WORD	wBias;	selected detector Bias mode	
WORD	wUgComp;	selected detector compensation	
WORD	wCameratype;	Detector type value defines the supported features (e.g. 1 support Binning, 2 supports Binning and special TriggerModes) (refer to Table 46: Detector Options Overview for more details)	
WORD	wFrameCnt;	Internal Frame counter of the detector	
WORD	wBinningMode;	selected detector binning mode:	
		Cameratype 1: BitNr set 0 active – default no binning 1 active - 2x2 Binning	Cameratype >=2: Refer to Table 35: Overview of the detector binning modes
WORD	wRealInttime_milliSec;	measured integration time of actual frame (millisec)	
WORD	wRealInttime_microSec;	measured integration time of actual frame (millisec)	

WORD	wStatus;	detector status word:
		
		Bit 0: 0 - OK 1 – Trigger lost
		
		Bit 1-3 Triggermode : Value
		
		  0: Data Delivered on Demand
		
		  1: Data Delivered on Demand with clearance scan
		
		  2: Linewise (Start/Stop)
		
		  3: Framewise (default)
		
		  4: AutoTrigger
		
		Bit 4: reserved
		
		Bit 5: DAC Error Code Bit 0
		
		Bit 6: DAC Error Code Bit 1
		
		Bit 7: DAC Error Code Bit 2
		
		Bit 8: Trigger Switch Current Setting 0 internal / 1 external
		
		Bit 12 : ADC readout 0: 14 bit 1: 16 bit	

WORD	**wCommand1**	reserved	
WORD	**wCommand2**	reserved	
WORD	**wCommand3**	reserved	
WORD	**wCommand4**	reserved	
WORD	**wDummy**	Contains the revision of the promid (wPROMID)	

Table 58: Description of the CHwHeaderInfoEx structure

REFERENCE MANUAL

7.2.50   Acquisition_GetHwHeaderInfoEx

This function **acquires** the frame header of the connected detector. If dwHeaderID in the CHwHeaderInfo structure is 14 (1621detectors) pInfoEx will retrieve the extended header, otherwise the structure will be filled with 0xFFFF.

UINT WINAPI Acquisition_GetHwHeaderInfoEx (
       HACQDESC   hAcqDesc,
       CHwHeaderInfo   *pInfo ,
       CHwHeaderInfoEx   *pInfoEx
       );

hAcqDesc
       Pointer to acquisition descriptor structure.
pInfo
       Pointer to Structure of type CHwHeaderInfo to retrieve the detector's hardware header.
pInfoEx
       Pointer to Structure of type CHwHeaderInfoEx to retrieve the detector's hardware
       header when available.
       Can be NULL.

Return values
If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.51   Acquisition_GetLatestFrameHeader

Use this function to retrieve the last acuiered frame header of the connected detector. If dwHeaderID in the CHwHeaderInfo structure is 14 pInfoEx will retrieve the extended header, otherwise the structure will be filled with 0xFFFF.

UINT WINAPI Acquisition_GetLatestFrameHeader (
       HACQDESC   hAcqDesc,
       CHwHeaderInfo   *pInfo ,
       CHwHeaderInfoEx   *pInfoEx
       );

hAcqDesc
       Pointer to acquisition descriptor structure.
pInfo
       Pointer to Structure of type CHwHeaderInfo to retrieve the detector's hardware header.
pInfoEx
       Pointer to Structure of type CHwHeaderInfoEx to retrieve the detector's hardware
       header when available.
       Can be NULL.

Return values
If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.52   Acquisition_ResetFrameCnt

This function is used to set internal frame counter to zero.

Note: If this function is used during active acquisition the detector Readout time may be affected.

UINT WINAPI Acquisition_ResetFrameCnt (
      HACQDESC hAcqDesc,
      );

hAcqDesc

      Pointer to acquisition descriptor structure.

Return values

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.53   Acquisition_SetCameraBinningMode

Use this function to set the detectors binning mode.

UINT WINAPI Acquisition_SetCameraBinningMode (
      HACQDESC   hAcqDesc,
      WORD   wMode
      );

hAcqDesc

      Pointer to acquisition descriptor structure.

wMode

      Binning Mode to be set (Bitwise)

      Cameratype 1:

      Bit 0 active: no binning (default)

      Bit 1 active: 2x2 binning

      Cameratype 2:

      Refer to Table 35: Overview of the detector binning modes.

Return values

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.54   Acquisition_GetCameraBinningMode

Use this function to retrieve the detectors binning mode.

UINT WINAPI Acquisition_GetCameraBinningMode (
      HACQDESC   hAcqDesc,
      WORD   *wMode
      );

hAcqDesc

      Pointer to acquisition descriptor structure.

wMode

      Pointer to value to retrieve the actual binning mode.

Return values

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.55   Acquisition_SetCameraTriggerMode

Use this function to set the detectors trigger mode.

UINT WINAPI Acquisition_SetCameraTriggerMode (
      HACQDESC   hAcqDesc,
      WORD   wMode
      );

hAcqDesc

      Pointer to acquisition descriptor structure.

wMode

      Trigger Mode to be set

      0: Data Delivered on Demand

      1: Data Delivered on Demand without clearance scan

      2: Linewise (Start/Stop)

      3: Framewise

      4: AutoTrigger [7]

Return values

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

Remark:

In section Acquisition_SetFrameSyncTimeMode(..) is a code snipped as an example how to use the 'Data delivered on Demand' mode. The delay time for the DDD-mode can be set using Acquisition_SetFrameSyncTimeMode(..).

7.2.56   Acquisition_GetCameraTriggerMode

Use this function to retrieve the detectors trigger mode.

UINT WINAPI Acquisition_GetCameraTriggerMode (
      HACQDESC hAcqDesc,
      WORD *wMode
      );

hAcqDesc

      Pointer to acquisition descriptor structure.

wMode

      Pointer to value to retrieve the actual trigger mode.

Return values

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

---

[7] Only for XRpad detector series available

7.2.57 Acquisition_SetTriggerOutSignalOptions

This function defines the behavior of the '/TrigOut' - signal of the detector trigger connector.

UINT WINAPI Acquisition_SetTriggerOutSignalOptions(
    HACQDESC   hAcqDesc,
    unsigned short   usTiggerOutSignalMode,
    unsigned short   usEP_SeqLength,
    unsigned short   usEP_FirstBrightFrm,
    unsigned short   usEP_LastBrightFrm,
    unsigned short   usEP_Delay1,
    unsigned short   usEP_Delay2,
    unsigned short   usDDD_Delay,
    int   iTriggerOnRisingEdgeEnable,
    int   iSaveAsDefault
    );

hAcqDesc
    Pointer to acquisition descriptor structure.
usTriggerOutSignalMode
    Select one of the follwing Modes:
        0 - FRM_EN_PWM
        1 - FRM_EN_PWM_INV
        2 - EP
        3 - EP_INV
        4 - DDD_Pulse
        5 - DDD_Pulse_INV
        6 - GND
        7 - VCC
usEP_SeqLength,
    Defines the Sequence Length for EP mode
usEP_FirstBrightFrm
    Defines the First Frame in the Sequence in which the TrigOut Signal is activated (zero-based)
usEP_LastBrightFrm
    Defines the Last Frame in the Sequence in which the TrigOut Signal is activated (zero-based)
usEP_Delay1
    Defines the Delay1 [in ms] from Begin of Frame in the EP_Frames until the TrigOut Signal gets activated
usEP_Delay2
    Defines the Delay2 [in ms] from Begin of Frame in the EP_Frames until the TrigOut Signal gets deactivated
usDDD_Delay
    Additional Delay in DDD Mode
iTriggerOnRisingEdgeEnable
    0 - Trigger falling edge
    1 - trigger on rising edge
iSaveAsDefault
    1 - Save usTriggerOutSignalMode and iTriggerOnRisingEdgeEnable
    in EEPROM to be available after the next power cycle

Return values
If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.58 Acquisition_SetCameraROI

With this function you can select out of four (Camera Type 10) or eight (Camera Type 12) predefined sections of the detector's field of view to combine them to one adjacent section. For more information see 6.2 Sectional Readout.

UINT WINAPI Acquisition_SetCameraROI(
    HACQDESC   hAcqDesc,
    unsigned short   usActivateGrp
    );

hAcqDesc
    Pointer to acquisition descriptor structure.
usActivateGrp
    Bitwise Selection of Groups to Readout and Transmit. Values to be set for are:

Detector / Cameratype	Value$_2$	Value$_{10}$	First row	Last row
0822 xP / 10	0001	1	0	255
	0010	2	256	511
	0100	4	512	767
	1000	8	768	1023
	0011	3	0	511
	0110	6	256	767
	1100	12	512	1023
	1111	15	0	1023
1642 xP / 12	0000 0001	1	0	127
	0000 0010	2	128	255
	0000 0100	4	256	383
	0000 1000	8	384	511
	0001 0000	16	512	639
	0010 0000	32	640	767
	0100 0000	64	768	895
	1000 0000	128	896	1023
	0000 0011	3	0	255
	0000 0110	6	128	383
	0000 1100	12	256	511
	0001 1000	24	384	639
	0011 0000	48	512	767
	0110 0000	96	640	895
	1100 0000	192	768	1023
	0000 1111	15	0	511
	0001 1110	30	128	639
	0011 1100	60	256	767
	0111 1000	120	384	895
	1111 0000	240	512	1023
	1111 1111	255	0	1023

Table 59: Bit combination for setting up Sectional Readout (XRD 0822xP/XRD1642xP Detector)

Return values

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.59 Acquisition_GetCameraROI

This function returnes the currently activated Region of Interest in Sectional Readout Mode.

UINT WINAPI Acquisition_GetCameraROI(
    HACQDESC   hAcqDesc,
    unsigned short   *usActivateGrp
    );

hAcqDesc

    Pointer to acquisition descriptor structure.

wMode

    Pointer to unsigend short value representing bitwisely coded the Selection of Groups to for the ROI

Return values

If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.60   Acquisition_ ActivateServiceMode

This function activates the Service Mode, which means Service Data is written into acquired images. This makes it easy to provide images for support.

UINT WINAPI Acquisition_ActivateServiceMode(
    HACQDESC   hAcqDesc,
    BOOL   bActivate
    );

hAcqDesc
    Pointer to acquisition descriptor structure.
bActivate
    Show Service Data: bActivate = TRUE
    otherwise: bActivate = FALSE

Return values
If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.2.61   Acquisition_SetRotationAngle

This function activates the onboard image rotation angle on optical frame-grabbers for images up to 2048x2048 pixels.

UINT WINAPI Acquisition_SetRotationAngle(
    HACQDESC   hAcqDesc,
    long lRotAngle
    );

hAcqDesc
    Pointer to acquisition descriptor structure.
lRotAngle
    Rotation angle: must be 0, 90 or -90

Return values
If the function is successful it returns zero, otherwise an error code. To get extended information call Acquisition_GetErrorCode.

7.3 Additional functions for devices with Gigabit Ethernet Interface

7.3.1 Introduction / Debugging

Setting up an Ethernet detector is special in comparison to Frame Grabber-connected devices. When the connection is estabished the detector and the network adapter permanently exchange the so called 'keep alive' signal to signalize that the connectin is active. This happens even if no image acquisition is performed (and therefore no image data is transmissed). But if the 'keep alive' signal is not present anymore this means that the connection to the detector is lost and has to be renewed and re-initialized completely.

In terms of code debugging this might be a problem: as debugging means running through the code step by step while stopping all other activities, the 'keep alive' indication can quickly run into a time-out and is consequently lost. This makes debugging almost impossible as you will not do that in real time. One possible way is to keep the 'keep alive' signal alive: Run a tool which catches the signal and simulates it, so that the connection cannot get lost. This tool must be independent from the software you debug, so that it can normally process without stopping in between.

This tool *Debug_Aid.exe* is located on the XIS installation disc in the XISL\WINxx\GBIF folder. Just start it before you run your application in debug mode and it will do the rest to keep the connection alive.

7.3.2 Definitions

GBIF_IP_MAC_NAME_CHAR_ARRAY_LENGTH	16
GBIF_STRING_DATATYPE	unsigned char
HIS_GbIF_IP_STATIC	1
HIS_GbIF_IP_DHCP	2
HIS_GbIF_IP_LLA	4
HIS_GbIF_IP	1
HIS_GbIF_MAC	2
HIS_GbIF_NAME	3

7.3.3 GBIF_DEVICE_PARAM

Structure that stores the connection contents of a GbIF detector.

typedef struct
{
    GBIF_STRING_DATATYPE ucMacAddress[GBIF_IP_MAC_NAME_CHAR_ARRAY_LENGTH];
    GBIF_STRING_DATATYPE ucIP[GBIF_IP_MAC_NAME_CHAR_ARRAY_LENGTH];
    GBIF_STRING_DATATYPE ucSubnetMask[GBIF_IP_MAC_NAME_CHAR_ARRAY_LENGTH];
    GBIF_STRING_DATATYPE ucGateway[GBIF_IP_MAC_NAME_CHAR_ARRAY_LENGTH];
    GBIF_STRING_DATATYPE ucAdapterIP[GBIF_IP_MAC_NAME_CHAR_ARRAY_LENGTH];
    GBIF_STRING_DATATYPE ucAdapterMask[GBIF_IP_MAC_NAME_CHAR_ARRAY_LENGTH];
    DWORD dwIPCurrentBootOptions;
    char cManufacturerName[32];
    char cModelName[32];
    char cGBIFFirmwareVersion[32];
    char cDeviceName[16];
} GBIF_DEVICE_PARAM

Description of structure entries

GBIF_STRING_DATATYPE	ucMacAddress	Physical address of the sensor (canonical format)
GBIF_STRING_DATATYPE	ucIP	Assigned or stored static IP address of the sensor

GBIF_STRING_DATATYPE	ucSubnetMask	Assigned or stored subnet mask of the sensor
GBIF_STRING_DATATYPE	ucGateway	Assigned or stored gateway of the sensor
GBIF_STRING_DATATYPE	ucAdapterIP	IP address of the host device
GBIF_STRING_DATATYPE	ucAdapterMask	Subnet mask of the host device
DWORD	dwIPCurrentBootOptions	Flag identifying the type of connection bitwisely: HIS_GbIF_IP_STATIC – Use Static IP address stored in the sensor HIS_GbIF_IP_LLA - Sensor will propose a Local Link Address HIS_GbIF_IP_DHCP - Sensor will receive IP address by DHCP server
CHAR[32]	cManufacturerName	"PerkinElmer"
CHAR[32]	cModelName	"GBIF" or "XRpad"
CHAR[32]	cGBIFFirmwareVersion	Firmware version of the GBIF module
CHAR[16]	cDeviceName	Type and serial number of the sensor

Table 60: Description of the GBIF_DEVICE_PARAM structure

7.3.4   Acquisition_GbIF_Init

The function **Acquisition_GbIF_Init** initializes the Ethernet connected sensors (XRpad and xx22) and the corresponding drivers. It prepares acquisition threads, defines callback functions to react on acquisition status changes: Furthermore it tests for sufficient memory space for DMA (direct memory access) and enables hardware interrupts if this modus is demanded.

UINT WINAPI Acquisition_GbIF_Init(
        HACQDESC   *phAcqDesc,
        int   nChannelNr,
        BOOL   bEnableIRQ,
        UINT   uiRows,
        UINT   uiColumns,
        BOOL   bSelfInit,
        BOOL   bAlwaysOpen,
        Long   lInitType,
        GBIF_STRING_DATATYPE* cAddress
        );

phAcqDesc

    Handle of a structure that contains all needed parameters for acquisition (HACQDESC).
    If you call Acquisition_GbIF_Init the first time set hAcqDesc to NULL, in subsequent
    calls use the former returned value.

nChannelNr

    This parameter defines a number to refer to the initialized device. For each GbIF sensor
    to be initialized an individual channel number has to be assigned. If you try to access
    multiple sensors using the same channel number, only the first one will be successfully
    initialized.

bEnableIRQ

    To run the acquisition in polling mode set this parameter to zero.
    To enable hardware interrupts set the parameter to one.

uiRows, uiColumns

    Number rows and columns of the sensor.

bSelfInit

    If *bSelfInit* is set to true the function retrieves the detector parameters (Rows,
    Columns,SortFlags) automatically.
    If *bSelfInit* is set to false the configuration parameters supplied by *Rows, Columns,*
    dwSortFlags are used.

bAlwaysOpen

If this parameter is TRUE the XISL is capturing the requested communication port regardless if this port is already opened by other processes running on the system. The use of this option is only recommended in debug versions of your applications because it is not possible to free all system resources allocated by another process.

lInitType

To identify of which type the parameter *cAddress* is, lInitType can have the following values:

HIS_GbIF_IP	The sensor is opend using the IP-Address passed in cAddress
HIS_GbIF_MAC	The sensor is opend using the MAC-Adress passed in cAddress
HIS_GbIF_NAME	The sensor is opend using the Detector-Name passed in cAddress

All values are defined within the file *acq.h*. IP-Address, MAC-Address and Detector Name can be retrieved by Acquisition_GbIF_GetDeviceList

cAddress

Address (array of 16 characters) to open the specified board. It can represent the MAC address, IP address or device name of the sensor.

Return Value

If the initialization is successful zero is returned, otherwise the return value is greater. Call Acquisition_GetErrorCode to get extended information.

7.3.5　Acquisition_GbIF_GetDeviceCnt

This function retrieves the total number of sensors found in the network by a network broadcast. If more than one network adapter are installed, a broadcast will be performed on all of them.

UINT WINAPI Acquisition_GbIF_GetDeviceCnt(
long　*plNrOfboards
);

plNrOfboards

Retrieves the number of sensors found in the network broadcasting.

Return Value

If the initialization is successful zero is returned, otherwise the return value is greater. Call Acquisition_GetErrorCode to get extended information.

7.3.6　Acquisition_GbIF_GetDeviceList

This function retrieves a list of parameter structures for all GbIF detector devices found by a network broadcast. If multiple network adapters are used in the host system, all of them are checked for connected GbIF detectors.

UINT WINAPI Acquisition_GbIF_GetDeviceList(
GBIF_DEVICE_PARAM *pGBIF_DEVICE_PARAM,
int　nDeviceCnt
);

pGBIF_DEVICE_PARAM

Returns a list of *GBIF_DEVICE_PARAM* elements, with *nDeviceCnt* entries. For that there has to be memory allocated of the size *nDeviceCnt*sizeof(GBIF_DEVICE_PARAM)* before calling the function.

nDeviceCnt

is the number of devices, which are found within the network. This parameter has to be passed to the function. It can be retrieved by calling *Acquisition_GbIF_GetDeviceCnt*.

Return Value

> If the initialization is successful zero is returned, otherwise the return value is greater.
> Call Acquisition_GetErrorCode to get extended information.

7.3.7   Acquisition_GbIF_GetDevice

This function retrieves the device parameter struct for the detector specified by the the passed address.

UINT WINAPI Acquisition_GbIF_GetDevice(
    GBIF_STRING_DATATYPE *ucAddress,
    DWORD   dwAddressType,
    GBIF_DEVICE_PARAM   *pDevice
    );

ucAddress

> Address (array of 16 characters) to open the specified board. It can represent the MAC address, IP address or device name of the sensor.

dwAddressType

> To identify of which type the parameter ucAddress is, lInitType
>
> can have the following values:

HIS_GbIF_IP	The sensor is selected by the IP-Address passed in cAddress
HIS_GbIF_MAC	The sensor is selected by the MAC-Adress passed in cAddress
HIS_GbIF_NAME	The sensor is selected by the Detector-Name passed in cAddress

> All values are defined within the file *acq.h*. IP-Address, MAC-Address and Detector Name can be retrieved by Acquisition_GbIF_GetDeviceList.

pDevice

> Pointer to structure which retrieves attributes and configuration of the device defined by *ucAddress.*

Return Value

> If the initialization is successful zero is returned, otherwise the return value is greater.
> Call Acquisition_GetErrorCode to get extended information.

7.3.8   Acquisition_GbIF_GetDeviceParams

This function retrieves the device parameters of a board that has already been opened.

UINT WINAPI Acquisition_GbIF_GetDeviceParams(
    HACQDESC   hAcqDesc,
    GBIF_DEVICE_PARAM *pDevice
    );

hAcqDesc

> Pointer to acquisition descriptor structure

pDevice

> Pointer to structure which retrieves attributes and configuration of the device defined by hAcqDesc.

7.3.9   Acquisition_GbIF_SetConnectionSettings

This function provides the parameters to configure how the detector connects to the network adapter. This function in not available for the XRpad.

UINT WINAPI Acquisition_GbIF_SetConnectionSettings(
      GBIF_STRING_DATATYPE   *cMAC,
      unsigned long   uiBootOptions,
      GBIF_STRING_DATATYPE   *cDefIP,
      GBIF_STRING_DATATYPE   *cDefSubNetMask,
      GBIF_STRING_DATATYPE   *cStdGateway
      );

cMAC
      MAC address of the device to be configured.
uiBootOptions
      OR-able flag to set the type of connection bitwisely:
            HIS_GbIF_IP_STATIC – Use Static IP address stored in the sensor
            HIS_GbIF_IP_LLA - Sensor will propose a Local Link Address
            HIS_GbIF_IP_DHCP - Sensor will receive IP address by DHCP server
cDefIP
      In case *uiBootOptions* is equal to HIS_GbIF_IP_STATIC, *cDefIP* can be used to set a new value as static IP. For that, *cDefIP* has to contain an IP address when calling the function.
cDefSubNetMask
      In case *uiBootOptions* is equal to HIS_GbIF_IP_STATIC, *cDefSubNetMask* can be used to set a new value as static IP. For that, *cDefSubNetMask* has to contain an IP address when calling the function.
cStdGateway
      In case *uiBootOptions* is equal to HIS_GbIF_IP_STATIC, *cStdGateway* can be used to set a new value as static IP. For that, *cStdGateway* has to contain an IP address when calling the function. When *cStdGateway* is zero the detector will be able to be initialized by Acquisition_EnumSensors(..)

Return Value
      If the initialization is successful zero is returned, otherwise the return value is greater. Call Acquisition_GetErrorCode to get extended information.

7.3.10 Acquisition_GbIF_GetConnectionSettings

This function retrieves the connection parameters of a GbIF detector.

UINT WINAPI Acquisition_GbIF_GetConnectionSettings(
        GBIF_STRING_DATATYPE   *ucMAC,
        unsigned long   *uiBootOptions,
        GBIF_STRING_DATATYPE   *ucDefIP,
        GBIF_STRING_DATATYPE   *ucDefSubNetMask,
        GBIF_STRING_DATATYPE   *ucStdGateway
        );

ucMAC
    MAC address of the device having the requested settings.
uiBootOptions
    OR-able flag indicating the type of connection bitwisely:
      HIS_GbIF_IP_STATIC – Use Static IP address stored in the sensor
      HIS_GbIF_IP_LLA - Sensor will propose a Local Link Address
      HIS_GbIF_IP_DHCP - Sensor will receive IP address by DHCP server
ucDefIP
    Retrieves the IP address the device is connected with.
ucDefSubNetMask
    Retrieves the Subnet the device is in.
ucStdGateway
    Retrieves the Standard Gateway of the connection to the device.

Return Value
    If the initialization is successful zero is returned, otherwise the return value is greater.
    Call Acquisition_GetErrorCode to get extended information.

7.3.11 Acquisition_GbIF_ForceIP

To configure the device it can be helpful to force the device temporarily to connect with a certain IP _Adress, usually one out of the same subnet and with the same StdGateway like the network card of your computer system. With restart of the detector the device will loose the temporary IP and behave as configured (e.g. IP per DHCP or LLA). This function in not available for the XRpad.

UINT WINAPI Acquisition_GbIF_ForceIP(
        GBIF_STRING_DATATYPE   *cMAC,
        GBIF_STRING_DATATYPE   *cDefIP,
        GBIF_STRING_DATATYPE   *cDefSubNetMask,
        GBIF_STRING_DATATYPE   *cStdGateway
        );

cMAC
    MAC address of the device to retrieve a temporary IP.
cDefIP
    Temporary IP
cDefSubNetMask
    Temporary Subnet Mask
cStdGateway
    Temporary Gateway

7.3.12   Acquisition_GbIF_SetPacketDelay

The Inter-Packet Delay can be set flexibly to balance out the workload of the IP connection between detector and network adapter. It is recommended to be configured depending on the network load. The value can be retrieved by calling *Acquisition_GbIF_CheckNetworkSpeed* (7.3.14).

UINT WINAPI Acquisition_GbIF_SetPacketDelay(
       HACQDESC   hAcqDesc,
       long   lPacketdelay
       );

hAcqDesc
       Pointer to acquisition descriptor structure
lPacketdelay
       Value for Inter-Packet Delay, which is to be set in the unit TICKS.

Return Value
       If the initialization is successful zero is returned, otherwise the return value is greater.
       Call Acquisition_GetErrorCode to get extended information.

7.3.13   Acquisition_GbIF_GetPacketDelay

Retrieve the InterPacket Delay, which is set for the current data connection.

UINT WINAPI Acquisition_GbIF_GetPacketDelay(
       HACQDESC   hAcqDesc,
       long   *lPacketdelay
       );

hAcqDesc
       Pointer to acquisition descriptor structure
lPacketdelay
       Retrieves the currently set value for Inter-Packet Delay

Return Value
       If the initialization is successful zero is returned, otherwise the return value is greater.
       Call Acquisition_GetErrorCode to get extended information.

7.3.14　　　　Acquisition_GbIF_CheckNetworkSpeed

This function determines which Detector Timing and Packetdelay can be set for the current system and network configuration. Please note that this function is intended for one detector only. If more than one detector is connected to the network adapter (detectors in LAN), the parameter lMaxNetworkPercent must divided by the number of connected sensors. Since the network load might change during operation this function cannot guarantee optimal performance. Use Acquisition_GbIF_SetPacketDelay and Acquisition_SetCameraMode(..) to apply the determined settings. For XRpad detectors a fixed value is returned for LAN/WLAN transmission. (80% network load are recommended)

UINT WINAPI Acquisition_GbIF_CheckNetworkSpeed(
    HACQDESC   hAcqDesc,
    WORD   *wTiming,
    long   *lPacketDelay,
    long   lMaxNetworkLoadPercent
    );

hAcqDesc
    Pointer to acquisition descriptor structure
wTiming
    Pointer to sugested Free Running Timing for actual System Performance
lPacketDelay
    Pointer to calculated max InterPacketDelay for suggested timing
lMaxNetworkPercent
    Percentage of Network load for which the Packet Delay shall be cheked.
    To allow a stable data transmissions with some space for packet resend select~80 percent

Return Value
    If the initialization is successful zero is returned, otherwise the return value is greater.
    Call Acquisition_GetErrorCode to get extended information.

7.3.15   GBIF_Detector_Properties

Structure to retrieve the properties of the connected GbIF detector.

typedef struct
{
   char cDetectorType[32];
   char cManufacturingDate[8];
   char cPlaceOfManufacture[8];
   char cDummy[80];
} GBIF_Detector_Properties

Description of structure entries

CHAR[32]	cDetectorType	Detector type description
CHAR[32]	cManufacturingDate	Manufacturing date (mm-yyyy)
CHAR[8]	cPlaceOfManufacture	Place of manufacturing
CHAR[80]	cDummy	not used

Table 61: Description of the GBIF_Detector_Properties structure

7.3.16   Acquisition_GbIF_GetDetectorProperties

This function fills the GBIF_Detector_Properties structure, which contains permanently stored information of the connected device.

UINT WINAPI Acquisition_GBIF_GetDetectorProperties(
    HACQDESC   hAcqDesc,
    GBIF_Detector_Properties *pDetectorProperties
    );

hAcqDesc

    Pointer to acquisition descriptor structure

pDetectorProperties

    Pointer to DetectorProperties structure. The memory for the object must be allocated before calling this function.

Return Value

    If the initialization is successful zero is returned, otherwise the return value is greater. Call Acquisition_GetErrorCode to get extended information.

7.3.17   Acquisition_GbIF_GetFilterDrvState

This function returns the status of the GbIF filter driver (if installed) otherwise an Error Code.

UINT WINAPI Acquisition_GbIF_GetFilterDrvState (
    HACQDESC hAcqDesc
    );

hAcqDesc

    Pointer to acquisition descriptor structure

Return Value:

    If the initialization is successful it returns the status of the Filter driver.
    1 for active, -1 for disabled / not installed
    If the function is not included in gbif.dll or the HACQDESC is not valid an Error Code is returned.

7.4 Additional functions for XRpad Detectors

7.4.1 Acquisition_wpe_GetVersion

This function returns the version of the used wpe200 library if available and loadable.

UINT WINAPI Acquisition_wpe_GetVersion(
      int * major,
      int * minor,
      int * release,
      int * build);

Return Value:
      HIS_ALL_OK function successfull otherwise an error code

7.4.2 Acquisition_wpe_getAvailableSystems

This function sends a networkbroadcast and retrieves all available XRpad detectors in the Network.

HIS_RETURN Acquisition_wpe_getAvailableSystems(
      struct discoveryReply * reply,
      int * numDevices,
      int timeout,
      int port);

Please refer to acq.h for the parameter definitions.
The user has to allocate the discoveryReply array before calling the function.

- numDevices has to contain the number of allcoted structures and will contain the retieved number of devices after the call. An array length of 10 is recommended
- Timeout and port must be 0 to use the predefined default values

Return Value:
      HIS_ALL_OK function successfull otherwise an error code

7.4.3 Acquisition_wpe_GetNetworkConfigs

This function retieves the Network settings of a specific Detector defined by its ip address.

HIS_RETURN Acquisition_wpe_GetNetworkConfigs(
      const char * ipAddress,
      struct networkConfiguration * configs,
      int * arrayLength,
      int * activeConfig);

Please refer to acq.h for the parameter definitions.

- The user has to allocate the networkConfiguration array before calling the function.
- arrayLength has to contain the number of allocated structures and will contain the retieved number of devices after the call. An array length of 10 is recommended
- activeConfig will contain the current activated configuration

Return Value:
      HIS_ALL_OK function successfull otherwise an error code

Acquisition_wpe_ActivateNetworkConfig

This function activates the selecte network configuration of the detector

HIS_RETURN Acquisition_wpe_ActivateNetworkConfig(**const char** * ipAddress, **int** configIndex);

Return Value:

       HIS_ALL_OK function successfull otherwise an error code

7.4.4   Acquisition_GetTriggerOutStatus

This function retrieves the triggerout status from detector

HIS_RETURN Acquisition_GetTriggerOutStatus(
      HACQDESC hAcqDesc,
      int * iTriggerStatus);

hAcqDesc

       Pointer to acquisition descriptor structure

iTriggerStatus *

       Pointer to value to retrieve the actual triggerout status.

       -1: Error, probably timeout, 0: triger low, 1: triger high

Return Value:

       HIS_ALL_OK trigger status could be retrieved otherwise an error code

7.4.5   Acquisition_wpe_ReadCameraRegisters

This function retrieves the Statusregister of a specific Detector definded by its ip address.

HIS_RETURN Acquisition_wpe_ReadCameraRegisters(
      const char * ipAddress,
      unsigned long * buffer);

Please refer to acq.h for the parameter definitions.
-    The user has to allocate the buffer for the register first
-    Buffer size has to be **EPC_REGISTER_LENGTH**
-    Buffer will contain Status Register as struct **EPC_REGISTER**

Return Value:

       HIS_ALL_OK status register could be retrieved otherwise an error code

typedef struct
{
      DWORD   year;
      DWORD   month;
      DWORD   day;
      DWORD   hour;
      DWORD   minute;
      DWORD   second;
} **RTC_STRUCT**;

DWORD	year;	The Year, for example 2013
DWORD	month;	The Month, for example May will result in 5
DWORD	day;	
DWORD	hour;	
DWORD	minute;	
DWORD	second;	

```
typedef struct
{
        DWORD           status;
        DWORD           serial_no;
        DWORD           cycle_count;
        DWORD           temperature;
        DWORD           voltage;
        DWORD           current;
        DWORD           capacity;
        DWORD           energy;
        DWORD           charge;
} DETECTOR_BATTERY;
```

DWORD	status;	Databit 0: present/not present, Databit 1: charging/not charging
DWORD	serial_no;	
DWORD	cycle_count;	
DWORD	temperature;	e.g. 2510 for 25.1 C
DWORD	voltage;	In mV
DWORD	current;	In mA (positive or negative)
DWORD	capacity;	In %
DWORD	energy;	In mWh
DWORD	charge;	In mAh

```
typedef struct
{
        DWORD           version;
        DWORD           temperature_value[8];
        DWORD           temperature_warning_level[8];
        DWORD           temperature_error_level[8];
        RTC_STRUCT      rtc_value;
        DETECTOR_BATTERY battery;
        DWORD           power_state;
        DWORD           sdcard_state;
        DWORD           sdcard_usage;
        DWORD           active_network_config;
        DWORD           lan_status_register;
        DWORD           wlan_status_register;
        DWORD           signal_strength;
        DWORD           channel;
        DWORD           exam_flag;
        DWORD           spartan_id;
        CHwHeaderInfoEx spartan_register;
} EPC_REGISTER;
```

DWORD	version;	
DWORD	temperature_value[8];;	In 1/1000 C (e.g. 43000 for 43 C)
DWORD	temperature_warning_level;	In 1/1000 C (e.g. 43000 for 43 C)
DWORD	temperature_error_level;	In 1/1000 C (e.g. 43000 for 43 C)
DWORD	rtc_value;	Structure see RTC_STRUCT
DWORD	battery;	Structure see DETECTOR_BATTERY
DWORD	power_state;	D0: is mounted
DWORD	sdcard_usage;	In %
DWORD	active_network_config;	
DWORD	lan_status_register;	D0: LAN enabled, D1: LAN up, D2: LAN used for Image transfer
DWORD	wlan_status_register;	D0: WLAN enabled, D1: WLAN up, D2: WLAN used for Image transfer, D3: is accesspoitn (0 for station), D4: HT20 mode, D5: HT40+ mode, D6: HT40- mode
DWORD	signal_strength;	In dBm
DWORD	channel;	
DWORD	exam_flag;	Active while Image Aquistion ongoing
DWORD	spartan_id;	Reserved currently: 1
DWORD	spartan_register;	Contains initial Hardware Header

7.4.6   Acquisition_GetFTPFile

This function retrieves a file from the SD-Card to a given buffer.

HIS_RETURN   Acquisition_GetFTPFile(
     const char * ipAddress,
     char * filename,
     void ** databuffer,
     long * filesize);

Please refer to acq.h for the parameter definitions.
- The function will allocate the buffer size

Return Value:
     HIS_ALL_OK trigger status could be retrieved otherwise an error code

7.4.7   Acquisition_FreeFTPFileBuffer

This function frees the buffer retrieved by Acquisition_GetFTPFile.

HIS_RETURN   Acquisition_FreeFTPFileBuffer (
     void * databuffer
     );

Please refer to acq.h for the parameter definitions.
- The function will free the buffer

Return Value:
     HIS_ALL_OK trigger status could be retrieved otherwise an error code

7.4.8   Acquisition_SetFTPFile

This function stores a databuffer on the sd card.

HIS_RETURN   Acquisition_SetFTPFile(
     const char * ipAddress,
     char * filename,
     void * databuffer,
     long filesize);

Please refer to acq.h for the parameter definitions.
- The buffersize has to be provided

Return Value:
     HIS_ALL_OK trigger status could be retrieved otherwise an error code

7.4.9   Acquisition_wpe_GetErrorCode
This function retrieves the last error code from the wpe200.dll.

HIS_RETURN   Acquisition_GetErrorCode()

Return Value:
     Latest error code

7.4.10   Acquisition_wpe_GetErrorCodeEx

This function retrieves the last error string from the wpe200.dll.

HIS_RETURN Acquisition_GetErrorCodeEx(**char *** pBuffer
      long len);

char * pBuffer

      Pointer to a character array to receive the latest error message.

      (char[256] recommended).

long len

      length of the passed character array

Return Value:
HIS_ALL_OK when function call was successfull otherwise an error code

7.5   Demo application

The XISL demonstration application is a simple console application running under Windows OS. It demonstrates how to acquire data continuously, how to acquire a sequence, how to create correction files and how to acquire corrected images.

You can install the demo sourcecode and the demo application by selecting **XISL SDK (32bit Windows)** or **XISL SDK (64bit Windows)** during the XIS installation process.

Files:

The 32bit demonstration program was build with MS Visual C++ 6.0. The name of the project file is "XISL_Demo.dsp". If you want to use your own development environment insert XISL.lib in your project. The header file declaring the required function prototypes is named "Acq.h". The main application program source is called "main.c".

The 64bit demonstration program was build with MS Visual C++ .NET 2008. The name of the project file is " XISL_DEMO_X64.vcproj". If you want to use your own development environment insert XISL.lib in your project. The header file declaring the required function prototypes is named "Acq.h". The main application program source is called "main.c".

7.6   XISL error codes

The following table lists all error codes of the XISL and gives a short description of them. The symbolic names of the errors are defined in Acq.h".

value	symbolic name	meaning
0	HIS_ALL_OK	No error
1	HIS_ERROR_MEMORY	Memory couldn't allocated.
2	HIS_ERROR_BOARDINIT	Unable to initialize board.
3	HIS_ERROR_NODETECTOR	Got a time out for acquisition. May be no detector present.
4	HIS_ERROR_CORRBUFFER_INCOMPATIBLE	Your correction files didn't have a proper size.
5	HIS_ERROR_ACQ_ALREADY_RUNNING	Unable to initialize board or allocate DMA buffer because a acquisition is running.
6	HIS_ERROR_TIMEOUT	Got a time out from hardware.
7	HIS_ERROR_INVALIDACQDESC	Acquisition descriptor invalid
8	HIS_ERROR_VXDNOTFOUND	Unable to find VxD.
9	HIS_ERROR_VXDNOTOPEN	Unable to open VxD.
10	HIS_ERROR_VXDUNKNOWNERROR	Unknown error during VxD loading.
11	HIS_ERROR_VXDGETDMAADDR	VxD Error: GetDmaAddr failed.
12	HIS_ERROR_ACQABORT	An unexpected acquisition abort occurred.
13	HIS_ERROR_ACQUISITION	An error occurred during data acquisition.
14	HIS_ERROR_VXD_REGISTER_IRQ	Unable to register interrupt.
15	HIS_ERROR_VXD_REGISTER_STATADR	Register status address failed.
16	HIS_ERROR_GETOSVERSION	Getting version of operating system failed.
17	HIS_ERROR_SETFRMSYNC	Can't set frame sync.
18	HIS_ERROR_SETFRMSYNCMODE	Can't set frame sync mode.
19	HIS_ERROR_SETTIMERSYNC	Can't set timer sync.
20	HIS_ERROR_INVALID_FUNC_CALL	Function was called by another thread than Acquisition_Init.
21	HIS_ERROR_ABORTCURRFRAME	Aborting current frame failed
22	HIS_ERROR_GETHWHEADERINFO	Getting hardware header failed
23	HIS_ERROR_HWHEADER_INF	Hardware header is invalid
24	HIS_ERROR_SETLINETRIG_MODE	Setting line trigger mode failed
25	HIS_ERROR_WRITE_DATA	Writing data failed
26	HIS_ERROR_READ_DATA	Reading data failed
27	HIS_ERROR_SETBAUDRATE	Setting baud rate failed
28	HIS_ERROR_NODESC_AVAILABLE	No acquisition descriptor available
29	HIS_ERROR_BUFFERSPACE_NOT_SUFF	Buffer space not sufficient
30	HIS_ERROR_SETCAMERAMODE	Setting detector mode failed
31	HIS_ERROR_FRAME_INV	Frame invalid
32	HIS_ERROR_SLOW_SYSTEM	System to slow
33	HIS_ERROR_GET_NUM_BOARDS	Error during getting number of boards
34	HIS_ERROR_ALREADY_OPEN_BY_ANOTHER_ PROCESS	Communication channel already opened by another process
35	HIS_ERROR_CREATE_MEMORYMAPPING	Error creating memory mapped file

36	HIS_ERROR_VXD_REGISTER_DMA_ADDRESS	Error registering DMA address
37	HIS_ERROR_VXD_REGISTER_STAT_ADDR	Error registering static address
38	HIS_ERROR_VXD_UNMASK_IRQ	Unable to unmask interrupt
39	HIS_ERROR_LOADDRIVER	Unable to load driver
40	HIS_ERROR_FUNC_NOTIMPL	Function is not implemented
41	HIS_ERROR_MEMORY_MAPPING	Unable to map memory
42	HIS_ERROR_CREATE_MUTEX	Mutex couldn't created
43	HIS_ERROR_ACQ	Error during acquisition
44	HIS_ERROR_DESC_NOT_LOCAL	Acquisition descriptor is not local
45	HIS_ERROR_INVALID_PARAM	Invalid Parameter
46	HIS_ERROR_ABORT	Error during abort acquisition
47	HIS_ERROR_WRONGBOARDSELECT	The wrong board is selected
48	HIS_ERROR_WRONG_CAMERA_MODE	Change of Detector Mode during Acquisition
49	HIS_ERROR_AVERAGED_LOST	The number of images for frame grabber onboard averaging must be 2^n
50	HIS_ERROR_BAD_SORTING_PARAM	Parameter for (onboard) sorting not valid
51	HIS_ERROR_UNKNOWN_IP_MAC_NAME	Connection to GigE Detector cannot be opened due to invalid IP address / MAC / Detector name
52	HIS_ERROR_NO_BOARD_IN_SUBNET	No GigE Detector could be found
53	HIS_ERROR_UNABLE_TO_OPEN_BOARD	Unable to open connection to GigE Detector
54	HIS_ERROR_UNABLE_TO_CLOSE_BOARD	Unable to close connection to GigE Detector
55	HIS_ERROR_UNABLE_TO_ACCESS_DETECTOR_FLASH	Unable to access flash memory of GigE Detector
56	HIS_ERROR_HEADER_TIMEOUT	No frame header received with GigE Detector
57	HIS_ERROR_NO_PING_ACK	Reserved
58	HIS_ERROR_NR_OF_BOARDS_CHANGED	Number of boards within network changed during broadcast

Table 62: Description XISL Error Codes

7.6.1 Frame Grabber error codes

The following table lists error codes of the XISL/FrameGrabber and gives a short description of them.

value	symbolic name	meaning
Warnings:		
4	EL_W_WRONGREVISIONCRC	Wrong CRC in hardware revision EEPROM
3	EL_W_ACQWINDOWTOOBIG	Acquisition window too big for the detector selected; will be fit automatically
2	EL_W_INLUTINDEXTOOBIG	The requested entry of the input look-up table does not exist. Valid are 0..255
1	EL_W_HWALREADYOPENED	The hardware has been opened without subsequent close. This may indicate that another task uses the DLL already or the DLL was not closed properly by an aborted task which can be tolerated.
0	EL_UNKNOWNERROR	Unexpected Error
Errors:		
-1	EL_E_WRONGBOARDSELECT	Board select parameter in function el_OpenHW invalid.
-2	EL_E_HWNOTOPENED	Hardware has not been opened – call el_OpenHW prior to the offending call.
-3	EL_E_BIOSNOTCORRECT	PCI Bios may not be present. The BIOS call Find PCI Bios did not return correct values. This call verifies no hardware access yet, it checks only that the BIOS can handle PCI functions and that it complies with PCI rev. 2.0.
-4	EL_E_NOPCEYEFOUND	PC_EYE board could not be found on PCI. This indicates that the PCI Bios of the computer is not capable of finding the PC_EYE board. PC_EYE boards can be identified by the driver in a unique way (Find PCI device and software-readable signature string).
-5	EL_E_PCEYESYSTEMMEMORY	The driver allocates n MB at startup time of Windows (n set in SYSTEM.INI); this may have failed at Windows start and is detected only now. The computer may not have enough memory installed.
-6	EL_E_FRAMEBUFALLOC	Memory for the frame buffer in the requested size could not be allocated. Closing other applications may help.
-7	EL_E_CONTEXTNOTINIT	The driver-internal context structure must be initialized first by calling el_InitContext.
-8	EL_E_HWNOTINIT	Call el_InitHW before using other functions.
-9	EL_E_PITCHTOOSMALL	The acquisition pitch is smaller than the horizontal image size.
-10	EL_E_MEMORYALLOC	Internal memory allocation failed. Closing other applications may help.
-11	EL_E_WRONGDETECTORSELECT	An invalid detector input number (0..3) is given.
-12	EL_E_ACQWINDOWTOOBIG	The acquisition window is too big for the detector selected. May indicate not enough memory. See description of el_NewMemBuffer.
-13	EL_E_WRONGBOARDID	No board with this ID is open.
-21	EL_E_UNKNOWNACQMODE	A number for a non-existing acquisition mode is given.

-22	EL_E_FUNCNOTAVAILABLE	Not implemented yet or wrong function code.
-23	EL_E_ACQTIMEOUT	The driver waits a certain time (about 5 frame times) for the acquisition to finish. This error occurs when the detector is not connected or powered down.
-24	EL_E_INVALIDPARAMETER	Invalid function parameter supplied.
-25	EL_E_INVALIDPOINTER	Invalid pointer supplied.
-64	EL_E_WRONGDETECTOR	unknown detector
-300	EL_E_IRQNOTIMPLEMENTED	IRQs are not implemented on this platform.
-301	EL_E_IRQISENABLED	IRQ is enabled.
-302	EL_E_IRQNOTENABLED	IRQ is not enabled.
-303	EL_E_IRQNOTAVAILABLE	IRQs are not available, check correct driver load order.
-304	EL_E_IRQINVALIDEVENT	Invalid IRQ-event type, use one of 'EL_IRQ_...'.
-305	EL_E_IRQINVALIDBOOST	Invalid priority boost.
-306	EL_E_IRQOPENEVENT	Error opening event.
-307	EL_E_IRQINTERNALERROR	Internal error setting up IRQs.

Table 63: Description of the Frame Grabber Error Codes

8 Configuration through Webinterface

8.1 Login

Default Webaddress (LAN) : http://192.168.2.158/

Default Webaddress (WLAN) : http://192.168.22.1/
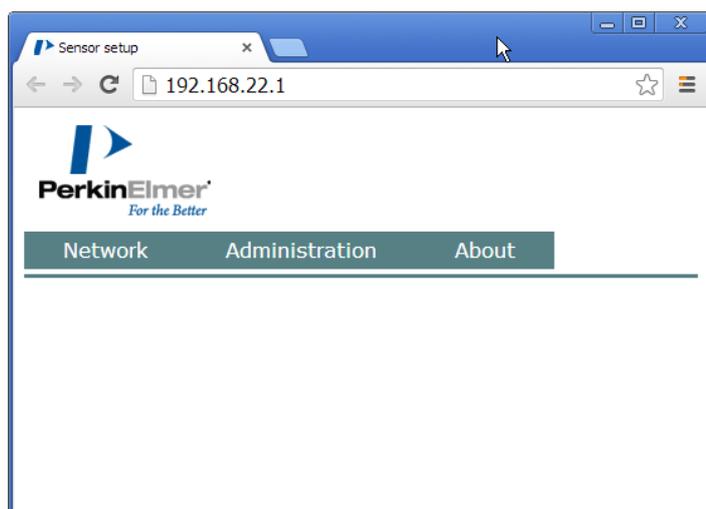
Figure 52 Start Screen Webserver
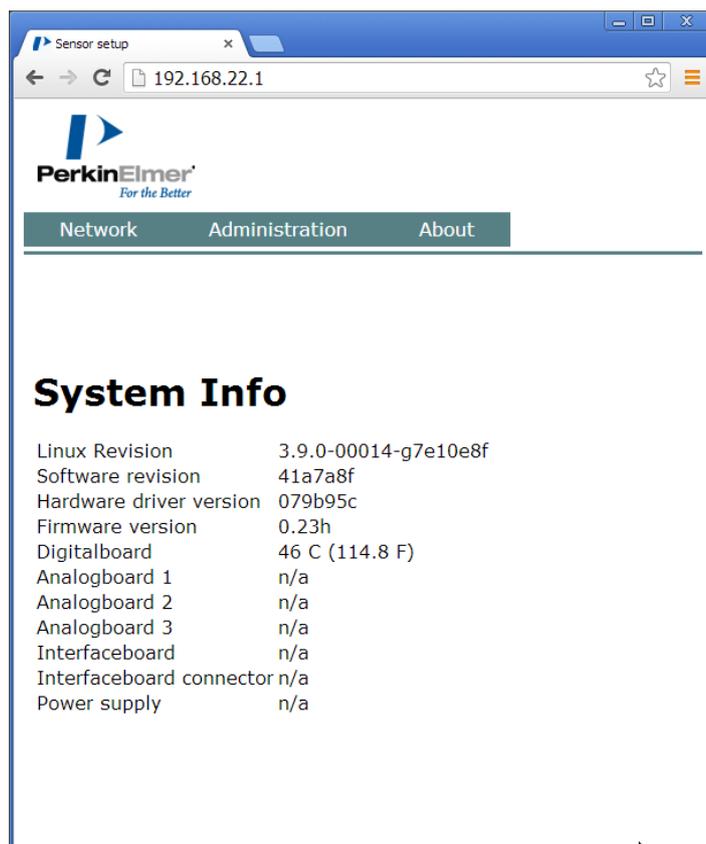
Figure 53 About Dialog / Version Information

The System Information page displays the software and firmware versions of the XRpad detector. These are important information for detector tracking.

PKI Firmware version will not be displayed here, but can be retrieved via Hardware Header. PLD version is not available to be retrieved.

onto set the Network settings username and a password are required. There are three levels of service levels available (Table 64).

Some of the Administration menu topics are for PKI usage only and require enhanced user rights. The next chapters are requiring the admin/cadmin-rights.

Figure 54 Authentication

Menu title	Menu entry	User Rights	Default password
Network			
	Network settings	admin	PerkinElmer
Administration			
	Language settings	admin	PerkinElmer
	System clock	admin	PerkinElmer
	Passwords	admin/cadmin	PerkinElmer / changeMe
	Wireless country code	cadmin	changeMe
	File upload	sadmin	Reserved for PKI only
	First system start up	sadmin	Reserved for PKI only
	Reset to factory settings	sadmin	Reserved for PKI only
About			
	System information	everybody	N/A

Table 64 Overview Webserver Menu

8.2   Network Settings

Figure 55 Overview Network Settings

Caution: Please ensure that there is only one Gateway and one Nameserver set!!

Don't set a Gateway and a Nameserver for WLAN and LAN.

8.3   Administration
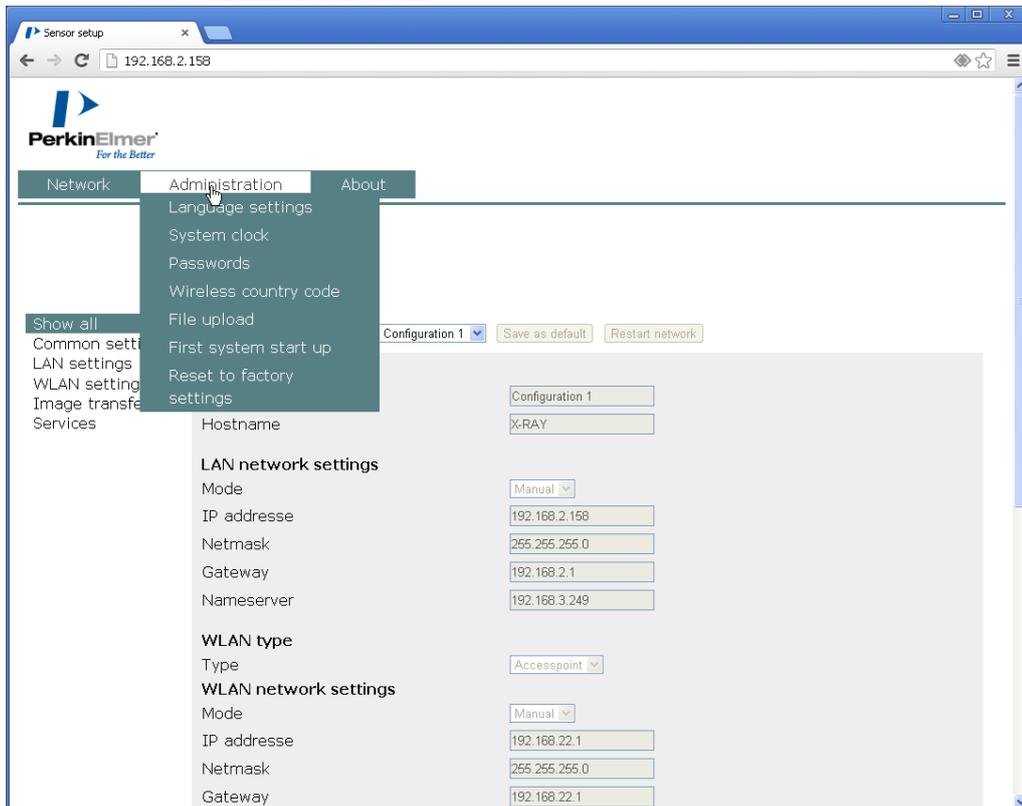
Figure 56 Menu Administration
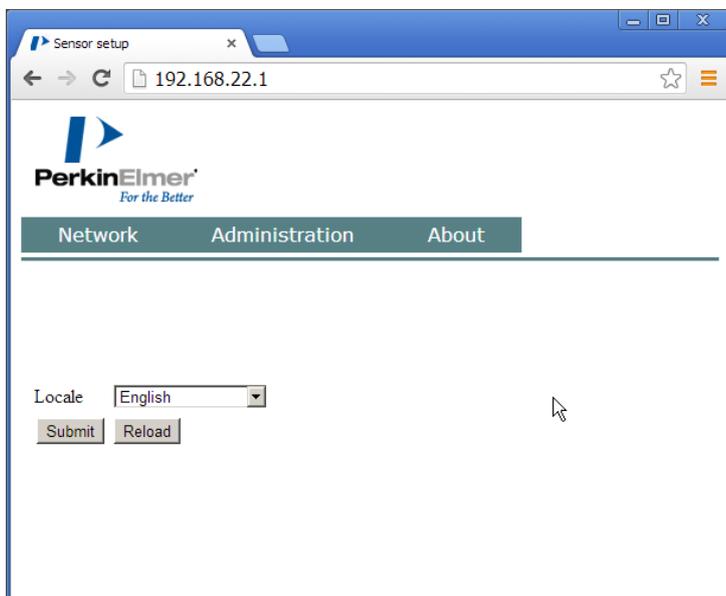
Figure 57 Language Settings for Webserver

Default language is determined by browser language settings which can be overwritten by selecting a different language in this dialog.
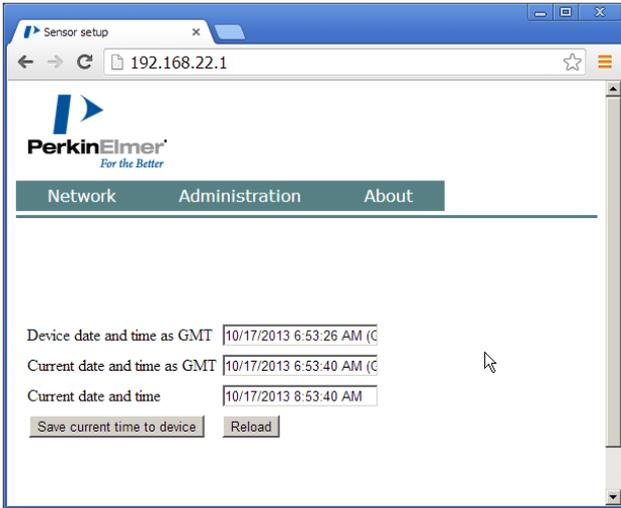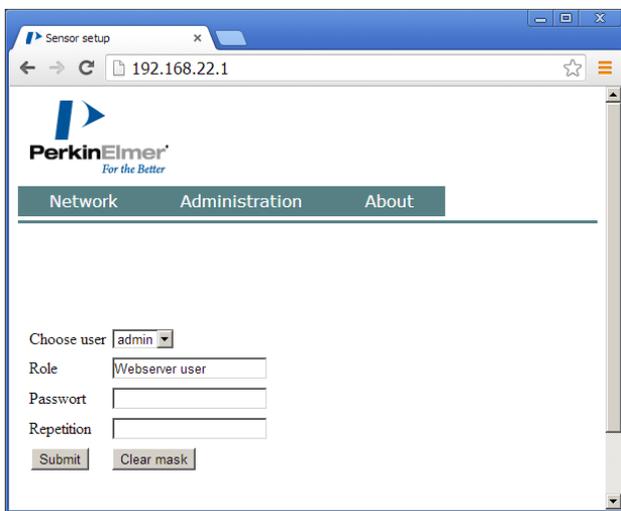
Figure 58 System Clock Dialog

Hint: the time is displayed in timezone GMT!

Figure 59 Change Passwords Dialog
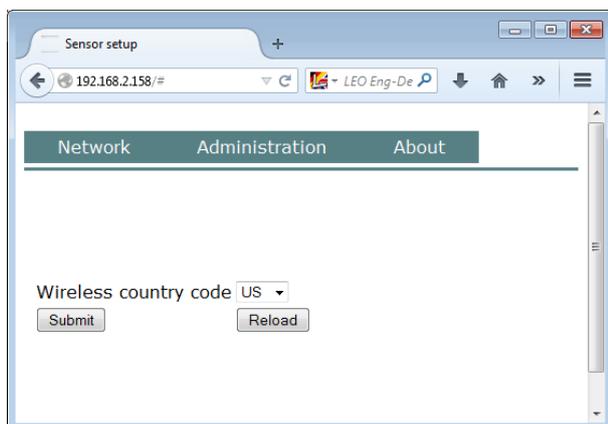
Figure 60 Administration – Wireless Country Code

This dialog sets the wireless country code which selects the accessablewireless channels. Please consider local regultatory requierements before shipping the device to a customer! cadmin rights are required to get access to this setting.

9   Detector SD-Card access (RO)

XRpad detector contains an internal SD-Card for data and software storage. Its possible to read and write to/from the detectors SD-Card via software or explorer.

The SD-Card can be read with an explorer from the ftp adress ftp://192.168.2.158 (default ip – or your active ip-adress). The explorer will show the content of the SD-Card (Read Only).

9.1   Login for SD-Card write access

To get write access to the SD-Card follow the steps below:
- open the explorer and enter ftp://192.168.2.158 (default ip – or your current active ip-adress)
- select "File" => "Login As…"
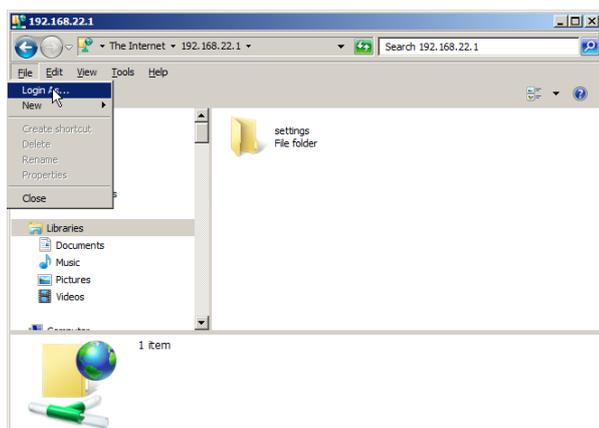
Figure 61 FTP SD-Card write access login

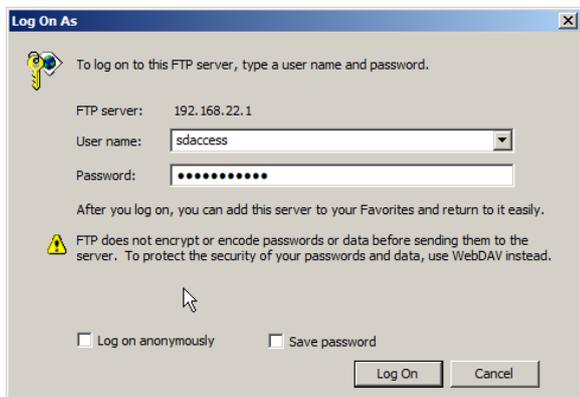- enter user name – "sdaccess" and password – "PerkinElmer" (default – changeable via web-server)

Figure 62 FTP SD-Card write access login dialog

- The explorer will show the content of the SD-Card for read/write from/to the SD-Card