Writing Applications with xiAPI

Default parameters

After camera is opened by xiOpenDevice the default camera parameters are set by API. The default parameters might be different in different API versions. In order to ensure that your application will have camera in expected state with any API version - please set all parameters expected by your application to required value.

---

xiAPI Functions

xiOpenDevice

Description:

This function initializes the device and returns a device handle.

Parameters:

- *DevId* - index of the device
- *hDevice* - handle to device

C Prototype:

XI_RETURN xiOpenDevice(IN DWORD DevId, OUT PHANDLE * hDevice);

Note: First call of this function enumerates all connected cameras. When the number of cameras is changing during the execution of a program, we recommend you to call xiGetNumberDevices function each time you call additional xiOpenDevice.

xiOpenDeviceBy

Description:

This function initializes the device and returns a device handle. Device is selected according to used enumerator.

Parameters:

- *sel* - select method to be used for camera selection
- *prm* - string to identify device
- *hDevice* - handle to device

C Prototype:

XI_RETURN __cdecl xiOpenDeviceBy(IN XI_OPEN_BY sel, IN const char* prm, OUT PHANDLE hDevice);

Corresponding Enumerator XI_OPEN_BY

type	representing value	result
XI_OPEN_BY_INST_PATH	0	Open camera by its hardware path
XI_OPEN_BY_SN	1	Open camera by its serial number
XI_OPEN_BY_USER_ID	2	Open camera by its custom user ID
XI_OPEN_BY_LOC_PATH	3	Open camera by its hardware location path

Note1: When the value of the parameter XI_PRM_DEVICE_USER_IDis changed and we want to open the camera with function xiOpenDeviceBy with the value XI_PRM_DEVICE_USER_ID, it is necessary to do a power cycle on this camera beforehand. This affects only cameras with USB data interface.

Note2: First call of this function enumerates all connected cameras. When the number of cameras is changing during the execution of a program, we recommend you to call xiGetNumberDevices function each time you call additional xiOpenDeviceBy.

xiCloseDevice

Description:

This function will un-initialize the specified device, closes its handle and releases allocated resources.

Parameters:

- *hDevice* - handle to device

C Prototype:

```
XI_RETURN xiCloseDevice(IN HANDLE hDevice);
```

xiGetNumberDevices

Description: This function enumerates all devices connected and returns the number of discovered devices. It is needed to be called before any other function of API is called by application.

Parameters:

- *pNumberDevices* - number of discovered devices

C Protopype:

```
XI_RETURN xiGetNumberDevices(OUT DWORD *pNumberDevices);
```

xiStartAcquisition

Description:

This function starts the data acquisition on the devices specified by the handle.

Parameters:

- *hDevice* - handle to device

C Prototype:

```
XI_RETURN xiStartAcquisition(IN HANDLE hDevice);
```

xiStopAcquisition

Description:

Ends the work cycle of the camera, stops data acquisition and deallocates internal image buffers.

Parameters:

- *hDevice* - handle to device

C Prototype:

```
XI_RETURN xiStopAcquisition(IN HANDLE hDevice);
```

xiGetImage

Description:

This function waits for next image is available at transport buffer. If available - it does all required image processing (unpack, sensor-defect-correction, demosaic) and fills image information to structure at img parameter. Image processing is not done if XI_FRM_TRANSPORT_DATA is selected. In this case the function just set pointer to transport-buffer without any processing.

Parameters:

- *hDevice* - handle to device
- *TimeOut* - time interval required to wait for the image (in milliseconds).
- *img* - Pointer to image info structure

Note: Allocation of buffers is influenced by buffering policy. See details of behavior at parameter XI_PRM_BUFFER_POLICY.

C Prototype:

XI_RETURN xiGetImage(IN HANDLE hDevice, IN DWORD TimeOut, INOUT XI_IMG * img);

The image structure XI_IMG description:

bytes	field name	description
4	size	Size of current structure on application side. When xiGetImage is called and size>=SIZE_XI_IMG_V2 then GPI_level, tsSec and tsUSec are filled.
ptr	bp	Pointer to data. (see Note1)
4	bp_size	Filled buffer size. (see Note2)
4	frm	Format of image data get from GetImage.
4	width	width of incoming image.
4	height	height of incoming image.
4	nframe	Frame number. On some cameras it is reset by exposure, gain, downsampling change, auto exposure (AEAG).
4	tsSec	Seconds part of image timestamp (see Note3).
4	tsUSec	Micro-seconds part image timestamp (see Note3). Range 0-999999 us.
4	GPI_level	Levels of digital inputs/outputs of the camera at time of exposure start/end (sample time and bits are specific for each camera model)
4	black_level	Black level of image (ONLY for MONO and RAW formats). (see Note4)
4	padding_x	Number of extra bytes provided at the end of each line to facilitate image alignment in buffers.
4	AbsoluteOffsetX	Horizontal offset of origin of sensor and buffer image first pixel.
4	AbsoluteOffsetY	Vertical offset of origin of sensor and buffer image first pixel.
4	transport_frm	Current format of pixels on transport layer.
x	img_desc	description of image areas and format.
4	DownsamplingX	Horizontal downsampling
4	DownsamplingY	Vertical downsampling

4	flags	description of XI_IMG.
4	exposure_time_us	Exposure time of this image in microseconds. (see Note5)
4	gain_db	Gain used for this image in deci-bells. (see Note6)
4	acq_nframe	Frame number. Reset only by acquisition start. NOT reset by change of exposure, gain, downsampling, auto exposure (AEAG).
4	image_user_data	(see Note7)
20	exposure_sub_times_us	(see Note8)
	data_saturation	Pixel value of saturation
4	wb_red	Red coefficient of white balance
4	wb_green	Green coefficient of white balance
4	wb_blue	Blue coefficient of white balance
4	lg_black_level	In case of multi gain channel readout, the black level low gain channel
4	hg_black_level	In case of multi gain channel readout, the black level high gain channel
4	lg_range	In case of multi gain channel readout, the valid range of low gain channel
4	hg_range	In case of multi gain channel readout, the valid range of high gain channel
4	gain_ratio	Gain ratio for dual-channel modes (high_gain_channel/low_gain_channel). Unitless.
4	fDownsamplingX	Horizontal downsampling
4	fDownsamplingY	Vertical downsampling
	color_filter_array	Mosaic of tiny color filters placed over the pixel sensors of an image sensor.
4	tof_phases_count	Number of phases of ToF sensor. E.g. 4 for four phases.
4	tof_phase_id	Current phase ID of ToF sensor data starting from 1
4	tof_multiple_phases_in_buffe	Is multiple phases in buffer (bp)
	data_sign_mode	Sign mode or signedness is a property of data.

Note1: If the buffer policy is set to XI_BP_UNSAFE, the bp is set to the buffer allocated by API. If set to XI_BP_SAFE, the data is copied to bp, which should be allocated by the application.

Note2: If the buffer policy is set to XI_BP_SAFE, xiGetImage fills this field with the current size of the received image data.

Note3: Depending on the camera family, the TimeStamp is represented as a counter:

・xiQ, xiD: 40-bit microsecond number - (overlaps after 305 hours)

・xiC, xiB, xiT, xiX: 64-bit 4 nanosecond number (overlaps after 2339 years)

This counter is converted to image header fields **tsSec** and **tsUSec.**

TimeStamp on **xiQ, xiD** is recorded at the start of Data Readout.

TimeStamp on **xiC, xiB, xiX, xiT** is recorded at the start of Exposure.

TimeStamp is NOT implemented on some cameras (e.g. xiMU - MU9), in which case the image header contains only a constant number instead of a valid TimeStamp.

Note4: xiQ cameras report calculated black level. We do not guarantee the accuracy of black level calculation when exposure time exceeds 50ms and/or gain is above 3dB.

Note5: Some camera models (MQ, MU) might report this exposure time earlier before exposure is applied to image. Cameras with IMX sensors (MC, MX, MT) report value measured by the FPGA - this value is systemically lower than the value returned by xiGetParam with XI_PRM_EXPOSURE parameter; difference is typically below 20 us.

Note6: Valid only for MQ, MD, and MR camera families with the following conditions:

・If a gain parameter is changed while the sensor is idle (not exposing nor reading out) the gain is valid for the next image.

・If a gain parameter is changed while the sensor is busy (exposing or reading out) and the 'direct_update' modifier is not used, the gain is valid for the next image.

・If a gain parameter is changed while the sensor is busy (exposing or reading out) and the 'direct_update' modifier is used, the gain value in the header might be incorrect after the change for the next 1-2 images. This is caused by the asynchronous setting of sensor registers and the frame acquisition process.

Note7: Available only on PCIe cameras (CB,MX). ImageUserData is controlled by a user application using ImageUserData or XI_PRM_IMAGE_USER_DATA parameter.

Note8: Array with five substitute exposure times in microseconds used by XI_TRG_SEL_MULTIPLE_EXPOSURES or hardware controlled HDR.

GPI_level-Sampling-in-Header

Digital Inputs are sampled on some cameras.

・MQ: at exposure end

・CB,MX,MC: at exposure start and exposure end

The samples are available after calling of xiGetImage using the parameters XI_PRM_GPI_LEVEL_AT_IMAGE_EXP_START, XI_PRM_GPI_LEVEL_AT_IMAGE_EXP_END. See example at the parameters description.

xiSetParam

Description:

This function configures device (see xiAPI Parameters below).

Parameters:

・*hDevice* - handle to device

・*prm* - parameter name string.

・*value* - value that should be set to parameter

・*size* - size of value

・*type* - data type of value

C Prototype:

```
XI_RETURN xiSetParam(IN HANDLE hDevice, IN CHAR * prm, IN VOID * value, IN DWORD size, IN XI_PRM_TYPE type);
```

xiGetDeviceInfoString

Description:

This function returns selected parameter of camera without opening it. It allows to quickly get information from each camera in multiple camera setups.

Parameters:

・*Devid* - index of the camera (same as on xiOpenDevice)

- *prm* - parameter name string
- *value* - pointer to result string
- *value_size* - size of string

C Prototype:

XIAPI XI_RETURN __cdecl xiGetDeviceInfoString(IN DWORD DevId, const char* prm, char* value, DWORD value_size);

Note: This function is capable to return only limited set of parameters:

- XI_PRM_DEVICE_SN
- XI_PRM_DEVICE_NAME
- XI_PRM_DEVICE_INSTANCE_PATH
- XI_PRM_DEVICE_LOCATION_PATH
- XI_PRM_DEVICE_TYPE

xiGetParam

Description:

This function returns parameters information (current value, minimum, maximum)(see xiAPI Parameters below).

Parameters:

- *hDevice* - handle to device
- *prm* - parameter name string
- *value* - value buffer where result will be stored
- *size* - size of value buffer
- *type* - expected value type

C Prototype:

XI_RETURN xiGetParam(IN HANDLE hDevice, IN CHAR * prm, IN VOID * value, INOUT DWORD * size, OUT XI_PRM_TYPE * type);