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 Parameters

Each parameter contains of:

- **Description:** Describing the parameter behavior
- **Type:** Data type used internally for modeling parameter
- **Default:** Default value, however it can differ between camera models.
- **Is invalidated by:** List of parameters. Changing any of the listed parameters may lead to the update of value or range (min, max, increment) of the respective parameter.
- **Usage:** Example of usage of this parameter in application (C code)

Basic

XI_PRM_EXPOSURE or "exposure"

Description: Current exposure time in microseconds. When parameter is set by xiSetParam the API checks the range. If it is within the range, it tries to find the closest settable value and set it. The actual value can be read by xiGetParam. E.g. Application set exposure time to 1000us, however closest possible value is 1029us (it is typically based on sensor line read-out period). This value is accessible over xiGetParam.

identifiers: SENSOR

Type: Float.

Default value: 1000.0

Usage:

```
xiSetParamInt(handle, XI_PRM_EXPOSURE, time_in_us);
```

XI_PRM_EXPOSURE_TIME_SELECTOR or "exposure_time_selector"

Description: Selector for Exposure parameter

Type: Enumerator.

Default value: XI_EXPOSURE_TIME_SELECTOR_COMMON

Usage:

```
int exposure_time_selector = 0;
```

xiGetParamInt(handle, XI_PRM_EXPOSURE_TIME_SELECTOR, &exposure_time_selector);
xiSetParamInt(handle, XI_PRM_EXPOSURE_TIME_SELECTOR, XI_EXPOSURE_TIME_SELECTOR_COMMON);

Value	Description
XI_EXPOSURE_TIME_SELECTOR_COMMON	Selects the common Exposure Time
XI_EXPOSURE_TIME_SELECTOR_GROUP1	Selects the common Exposure Time for pixel group 1 (for InterlineExposureMode)
XI_EXPOSURE_TIME_SELECTOR_GROUP2	Selects the common Exposure Time for pixel group 2 (for InterlineExposureMode)
XI_EXPOSURE_TIME_SELECTOR_DUAL_TRG_EXP_ZONE_1	Selects the Exposure Time for Zone 1 (for Dual Trigger Exposure feature)
XI_EXPOSURE_TIME_SELECTOR_DUAL_TRG_EXP_ZONE_2	Selects the Exposure Time for Zone 2 (for Dual Trigger Exposure feature)

XI_PRM_EXPOSURE_BURST_COUNT or "exposure_burst_count"

Description: Sets the number of times of exposure in one frame. To finish exposure burst change this parameter to 1 and exposure will be finished by next trigger. See more details in article Multiple exposures in one frame.

Note: This setting is valid only if the trigger selector is set to ExposureActive or ExposureStart.
Supported cameras: MC031xG-SY, MC050xG-SY, MC089xG-SY, MC124xG-SY, MX031xG-SY, MX050xG-SY, MX089xG-SY, MX124xG-SY, MT031xG-SY, MT050xG-SY

Type: Integer.

Default value: 1

Usage:

xiSetParamInt(handle, XI_PRM_TRG_SELECTOR, XI_TRG_SEL_EXPOSURE_ACTIVE);
xiSetParamInt(handle, XI_PRM_EXPOSURE_BURST_COUNT, 5)

XI_PRM_GAIN_SELECTOR or "gain_selector"

Description: Selects type of gain for XI_PRM_GAIN. On some cameras there is possibility to select analog or digital gain separately.

Selector XI_GAIN_SELECTOR_ALL is mapped on most cameras to analog gain.

Type: Enumerator.

Default value: XI_GAIN_SELECTOR_ANALOG_ALL

Usage:

xiSetParamInt(handle, XI_PRM_GAIN_SELECTOR, XI_GAIN_SELECTOR_ANALOG_ALL);

Value	Description
XI_GAIN_SELECTOR_ALL	Gain selector selects all channels. Implementation of gain type depends on camera.

XI_GAIN_SELECTOR_ANALOG_ALL	Gain selector selects all analog channels. This is available only on some cameras.
XI_GAIN_SELECTOR_DIGITAL_ALL	Gain selector selects all digital channels. This is available only on some cameras.
XI_GAIN_SELECTOR_ANALOG_TAP1	Gain selector selects tap 1. This is available only on some cameras.
XI_GAIN_SELECTOR_ANALOG_TAP2	Gain selector selects tap 2. This is available only on some cameras.
XI_GAIN_SELECTOR_ANALOG_TAP3	Gain selector selects tap 3. This is available only on some cameras.
XI_GAIN_SELECTOR_ANALOG_TAP4	Gain selector selects tap 4. This is available only on some cameras.
XI_GAIN_SELECTOR_ANALOG_N	First of two channels of programmable gain control (PGC) function - Gain setting of R, B pixels (North column analog gain). This is available only on some cameras.
XI_GAIN_SELECTOR_ANALOG_S	Second of two channels of programmable gain control (PGC) function - Gain setting of Gr, Gb pixels (South column analog gain). This is available only on some cameras.

XI_PRM_GAIN or "gain"

Description: Current gain in dB. When parameter is set by xiSetParam the API checks the range. If it is within the range, it tries to find the closest settable value and set it. The actual value can be read by xiGetParam. E.g. Application set gain to 1.3dB, however closest possible value is 1.35dB (analog gain is typically based on sensor PGA registers capabilities). This value is accessible over xiGetParam.

Type: Float.

Default value: 0.0

Is invalidated by: XI_PRM_DOWNSAMPLING_TYPE, XI_PRM_DOWNSAMPLING, XI_PRM_BINNING_VERTICAL, XI_PRM_BINNING_HORIZONTAL, XI_PRM_DECIMATION_VERTICAL, XI_PRM_DECIMATION_HORIZONTAL, XI_PRM_DP_PARAM_VALUE, XI_PRM_HDR, XI_PRM_SENSOR_DATA_BIT_DEPTH, XI_PRM_USER_SET_LOAD

Usage:

```
xiSetParamFloat(handle, XI_PRM_GAIN, gain_in_db);
```

XI_PRM_DOWNSAMPLING or "downsampling"

Description: Changes image resolution by binning or skipping. Parameter downsampling_type controls the mapping of sensor pixels to output data.

Note1: Downsampling can be changed only before an acquisition is started.

Note2: Changing this parameter will flush all images from the buffer queue.

Type: Enumerator.

Default value: 1

Usage:

```
xiSetParamInt(handle, XI_PRM_DOWNSAMPLING, XI_DWN_2x2);
```

Value	Description
XI_DWN_1x1	1 sensor pixel = 1 image pixel
XI_DWN_2x2	2x2 sensor pixels = 1 image pixel
XI_DWN_3x3	Downsampling 3x3.
XI_DWN_4x4	4x4 sensor pixels = 1 image pixel
XI_DWN_5x5	Downsampling 5x5.
XI_DWN_6x6	Downsampling 6x6.
XI_DWN_7x7	Downsampling 7x7.
XI_DWN_8x8	Downsampling 8x8.
XI_DWN_9x9	Downsampling 9x9.
XI_DWN_10x10	Downsampling 10x10.
XI_DWN_16x16	Downsampling 16x16.

XI_PRM_DOWNSAMPLING_TYPE or "downsampling_type"

Description: Changes image downsampling type (binning or skipping).
Note1: Changing this parameter will remove all images from the buffer queue.
Note2: Changing this parameter will remove all images from the buffer queue.
Type: Enumerator.
Default value: XI_BINNING
Usage:

xiSetParamInt(handle, XI_PRM_DOWNSAMPLING_TYPE, XI_SKIPPING);

Value	Description
XI_BINNING	pixels are interpolated - better image
XI_SKIPPING	pixels are skipped - higher frame rate

XI_PRM_TEST_PATTERN_GENERATOR_SELECTOR or "test_pattern_generator_selector"

Description: Selects Test Pattern Generator Engine.
Type: Enumerator.
Default value: XI_TESTPAT_GEN_SENSOR
Usage:

xiSetParamInt(handle, XI_PRM_TEST_PATTERN_GENERATOR_SELECTOR, XI_TESTPAT_GEN_SENSOR);

Value	Description
XI_TESTPAT_GEN_SENSOR	Sensor test pattern generator
XI_TESTPAT_GEN_FPGA	FPGA Test Pattern Generator
XI_TESTPAT_GEN_MCU	MCU Test Pattern Generator

XI_PRM_TEST_PATTERN or "test_pattern"

Description: Selects Test Pattern Type to be generated by the selected Generator Engine.

Type: Enumerator.

Default value: XI_TESTPAT_OFF

Is invalidated by: XI_PRM_TEST_PATTERN_GENERATOR_SELECTOR

Usage:

```
int test_pattern = 0;
xiGetParamInt(handle, XI_PRM_TEST_PATTERN, &test_pattern);
xiSetParamInt(handle, XI_PRM_TEST_PATTERN, XI_TESTPAT_OFF);
```

Value	Description
XI_TESTPAT_OFF	Testpattern turned off.
XI_TESTPAT_BLACK	Image is filled with darkest possible image.
XI_TESTPAT_WHITE	Image is filled with brightest possible image.
XI_TESTPAT_GREY_HORIZ_RAMP	Image is filled horizontally with an image that goes from the darkest possible value to the brightest.
XI_TESTPAT_GREY_VERT_RAMP	Image is filled vertically with an image that goes from the darkest possible value to the brightest.
XI_TESTPAT_GREY_HORIZ_RAMP_MOVING	Image is filled horizontally with an image that goes from the darkest possible value to the brightest and moves from left to right.
XI_TESTPAT_GREY_VERT_RAMP_MOVING	Image is filled vertically with an image that goes from the darkest possible value to the brightest and moves from left to right.
XI_TESTPAT_HORIZ_LINE_MOVING	A moving horizontal line is superimposed on the live image.
XI_TESTPAT_VERT_LINE_MOVING	A moving vertical line is superimposed on the live image.
XI_TESTPAT_COLOR_BAR	Image is filled with stripes of color including White, Black, Red, Green, Blue, Cyan, Magenta and Yellow.
XI_TESTPAT_FRAME_COUNTER	A frame counter is superimposed on the live image.
XI_TESTPAT_DEVICE_SPEC_COUNTER	128bit counter.

XI_PRM_IMAGE_DATA_FORMAT or "imgdataformat"

Description: Format of image data returned by function xiGetImage. In order to simplify the control of the camera from application - the xiAPI automatically changes selected camera parameters and Image Processing after setting of XI_PRM_IMAGE_DATA_FORMAT

In enumerators table second value in comment bar stands for one pixel data in memory [one_byte].

Note: Following parameters and Image Processing are controlled automatically by setting of XI_PRM_IMAGE_DATA_FORMAT:

Format: XI_MONO8 **Parameters controlled automatically:**

- XI_PRM_SENSOR_DATA_BIT_DEPTH = 8 (see Note1)
- XI_PRM_OUTPUT_DATA_BIT_DEPTH = 8 (see Note1)
- XI_PRM_OUTPUT_DATA_PACKING = OFF

Image Processing: enabled

Format: XI_RAW8 **Parameters controlled automatically:**

・XI_PRM_SENSOR_DATA_BIT_DEPTH = 8 (see Note1)

・XI_PRM_OUTPUT_DATA_BIT_DEPTH = 8 (see Note1)

・XI_PRM_OUTPUT_DATA_PACKING = OFF

Image Processing: disabled

Format: XI_MONO16 (see Note2) **Parameters controlled automatically:**

・XI_PRM_SENSOR_DATA_BIT_DEPTH = maximum

・XI_PRM_OUTPUT_DATA_BIT_DEPTH = SENSOR_DATA_BIT_DEPTH

・XI_PRM_OUTPUT_DATA_PACKING = ON (see Note1)

Image Processing: enabled

Format: XI_RAW16 (see Note2) **Parameters controlled automatically:**

・XI_PRM_SENSOR_DATA_BIT_DEPTH = maximum

・XI_PRM_OUTPUT_DATA_BIT_DEPTH = SENSOR_DATA_BIT_DEPTH

・XI_PRM_OUTPUT_DATA_PACKING = ON (see Note1)

Image Processing: disabled

Format: XI_RGB32, XI_RGB24, XI_RGB_PLANAR **Parameters controlled automatically:**

・XI_PRM_SENSOR_DATA_BIT_DEPTH = maximum

・XI_PRM_OUTPUT_DATA_BIT_DEPTH = SENSOR_DATA_BIT_DEPTH

・XI_PRM_OUTPUT_DATA_PACKING = ON (see Note1)

Image Processing: enabled

Note1: Only if camera implementation allows this mode.

Note2: For XI_RAW16, XI_MONO16 the parameter XI_PRM_IMAGE_DATA_BIT_DEPTH will be equal to XI_PRM_OUTPUT_DATA_BIT_DEPTH. For other formats the XI_PRM_IMAGE_DATA_BIT_DEPTH will be 8.

After changing of XI_PRM_IMAGE_DATA_FORMAT the image resolution () might change. Please check or set image resolution after changing of Image Data Format.

Note3: Bits alignment: Values are aligned to LSB.

・sensor bits per pixel: **10** >>> values in mode XI_RAW16: **0-1023**

・sensor bits per pixel: **12** >>> values in mode XI_RAW16: **0-4095**

・sensor bits per pixel: **14** >>> values in mode XI_RAW16: **0-16383**

Example: Camera produces 10 bits data and data format XI_RAW16bit is selected - each 16bit word (pixel) can contain values in range 0-1023.

Note4: For color modes XI_RGB32 and XI_RGB24 the image from sensor should be pre-processed. CPU load is higher in these modes. Setting this parameter will reset current region of interest. XI_RGB24 is being processed from the XI_RGB32 by removing the unused Alpha channel creating a slightly higher CPU load then the XI_RGB32 format.

Type: Enumerator.

Default value: XI_MONO8

Usage:

```
int imgdataformat = 0;
```

xiGetParamInt(handle, XI_PRM_IMAGE_DATA_FORMAT, &imgdataformat);

xiSetParamInt(handle, XI_PRM_IMAGE_DATA_FORMAT, XI_MONO8);

Value	Description
XI_MONO8	8 bits per pixel. [Intensity] (see Note5,Note6)
XI_MONO16	16 bits per pixel. [Intensity LSB] [Intensity MSB] (see Note5,Note6)
XI_RGB24	RGB data format. [Blue][Green][Red] (see Note5)
XI_RGB32	RGBA data format. [Blue][Green][Red][0] (see Note5)
XI_RGB_PLANAR	RGB planar data format. [Red][Red]...[Green][Green]...[Blue][Blue]... (see Note5)
XI_RAW8	8 bits per pixel raw data from sensor. [pixel byte] raw data from transport (camera output)
XI_RAW16	16 bits per pixel raw data from sensor. [pixel byte low] [pixel byte high] 16 bits (depacked) raw data
XI_FRM_TRANSPORT_DATA	Data from transport layer (e.g. packed). Depends on data on the transport layer (see Note7)
XI_RGB48	RGB data format. [Blue low byte][Blue high byte][Green low][Green high][Red low][Red high] (see Note5)
XI_RGB64	RGBA data format. [Blue low byte][Blue high byte][Green low][Green high][Red low][Red high][0][0] (Note5)
XI_RGB16_PLANAR	RGB16 planar data format
XI_RAW8X2	8 bits per pixel raw data from sensor(2 components in a row). [ch1 pixel byte] [ch2 pixel byte] 8 bits raw data from 2 channels (e.g. high gain and low gain channels of sCMOS cameras)
XI_RAW8X4	8 bits per pixel raw data from sensor(4 components in a row). [ch1 pixel byte [ch2 pixel byte] [ch3 pixel byte] [ch4 pixel byte] 8 bits raw data from 4 channels (e.g. sCMOS cameras)
XI_RAW16X2	16 bits per pixel raw data from sensor(2 components in a row). [ch1 pixel byte low] [ch1 pixel byte high] [ch2 pixel byte low] [ch2 pixel byte high] 16 bits (depacked) raw data from 2 channels (e.g. high gain and low gain channels of sCMOS cameras)
XI_RAW16X4	16 bits per pixel raw data from sensor(4 components in a row). [ch1 pixel byte low] [ch1 pixel byte high] [ch2 pixel byte low] [ch2 pixel byte high] [ch3 pixel byte low] [ch3 pixel byte high] [ch4 pixel byte low] [ch4 pixel byte high] 16 bits (depacked) raw data from 4 channels (e.g. sCMOS cameras)
XI_RAW32	32 bits per pixel raw data from sensor in integer format (LSB first). 4 bytes (LSB first) pixel (depacked) raw data
XI_RAW32FLOAT	32 bits per pixel raw data from sensor in single-precision floating point format. 4 bytes per pixel (depacked) raw data

Note5: Higher CPU processing is required when this mode is selected because color filter array processing is implemented on PC. This processing is serialized when multiple cameras is used at once. The most effective way to get data from camera is to use XI_RAW8, where no additional processing is done in API.

Note6: On monochromatic cameras the black level is not subtracted in XI_MONO8 and XI_MONO16 formats by Image Processing in xiAPI, so black level remains the same as in RAW format.

Note7: When using Transport Data Format, the Image Processing block from XiAPI Image Data Flow is skipped and therefore the Transport format is the most effective data format in terms of CPU and RAM usage.

XI_PRM_IMAGE_DATA_SIGN or "image_data_sign"

Description: Signedness of image data.

Type: Enumerator.

Default value: XI_DATA_SM_UNSIGNED

Is invalidated by: XI_PRM_IMAGE_DATA_FORMAT, XI_PRM_TOF_READOUT_MODE

Usage:

```
int image_data_sign = 0;
xiGetParamInt(handle, XI_PRM_IMAGE_DATA_SIGN, &image_data_signsizeof(value));
```

Value	Description
XI_DATA_SM_UNSIGNED	Unsigned if it can only represent non-negative numbers (zero or positive numbers).
XI_DATA_SM_SIGNED_2C	Signed if it can represent both positive and negative numbers (two's complement).
XI_DATA_SM_SIGNED_FLOATING	Signed floating point data type.

XI_PRM_SHUTTER_TYPE or "shutter_type"

Description: Type of sensor shutter.

Type: Enumerator.

Default value: XI_SHUTTER_GLOBAL

Is invalidated by: XI_PRM_TRG_SOURCE

Usage:

```
int shutter_type = 0;
xiGetParamInt(handle, XI_PRM_SHUTTER_TYPE, &shutter_type);
xiSetParamInt(handle, XI_PRM_SHUTTER_TYPE, XI_SHUTTER_GLOBAL);
```

Value	Description
XI_SHUTTER_GLOBAL	Sensor Global Shutter(CMOS sensor)
XI_SHUTTER_ROLLING	Sensor Electronic Rolling Shutter(CMOS sensor)
XI_SHUTTER_GLOBAL_RESET_RELEASE	Sensor Global Reset Release Shutter(CMOS sensor)

XI_PRM_SENSOR_TAPS or "sensor_taps"

Description: Set/get the number of taps used on sensor.

Type: Enumerator.

Default value: 1

Usage:

```
int sensor_taps = 0;
xiGetParamInt(handle, XI_PRM_SENSOR_TAPS, &sensor_taps);
xiSetParamInt(handle, XI_PRM_SENSOR_TAPS, XI_TAP_CNT_1);
```

Value	Description
XI_TAP_CNT_1	1 sensor tap selected.
XI_TAP_CNT_2	2 sensor taps selected.
XI_TAP_CNT_4	4 sensor taps selected.

XI_PRM_AEAG or "aeag"

Description: Automatic exposure/gain.

Type: Integer.

Default value: XI_OFF

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_AEAG, &value);
xiSetParamInt(handle, XI_PRM_AEAG, XI_ON);
```

XI_PRM_AEAG_ROI_OFFSET_X or "aeag_roi_offset_x"

Description: X offset of the area used for AEAG calculation. The sum of XI_PRM_AEAG_ROI_OFFSET_X and XI_PRM_AEAG_ROI_WIDTH must be equal or lower than the image resolution(width).

Type: Integer.

Default value: 0

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_AEAG_ROI_OFFSET_X, &value);
xiSetParamInt(handle, XI_PRM_AEAG_ROI_OFFSET_X, value);
```

XI_PRM_AEAG_ROI_OFFSET_Y or "aeag_roi_offset_y"

Description: Y offset of the area used for AEAG calculation. The sum of XI_PRM_AEAG_ROI_OFFSET_Y and XI_PRM_AEAG_ROI_HEIGHT must be equal or lower than the image resolution(height).

Type: Integer.

Default value: 0

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_AEAG_ROI_OFFSET_Y, &value);
```

xiSetParamInt(handle, XI_PRM_AEAG_ROI_OFFSET_Y, value);

XI_PRM_AEAG_ROI_WIDTH or "aeag_roi_width"

Description: width of the area used for AEAG calculation. The sum of XI_PRM_AEAG_ROI_OFFSET_X and XI_PRM_AEAG_ROI_WIDTH must be equal or lower than the image resolution(width).

Type: Integer.

Default value: Depends on the sensors resolution and downsampling.

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_AEAG_ROI_WIDTH, &value);
xiSetParamInt(handle, XI_PRM_AEAG_ROI_WIDTH, value);
```

XI_PRM_AEAG_ROI_HEIGHT or "aeag_roi_height"

Description: height of the area used for AEAG calculation. The sum of XI_PRM_AEAG_ROI_OFFSET_Y and XI_PRM_AEAG_ROI_HEIGHT must be equal or lower than the image resolution(height).

Type: Integer.

Default value: Depends on the sensors resolution and downsampling.

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_AEAG_ROI_HEIGHT, &value);
xiSetParamInt(handle, XI_PRM_AEAG_ROI_HEIGHT, value);
```

XI_PRM_SENS_DEFECTS_CORR_LIST_SELECTOR or "bpc_list_selector"

Description: Selector for current sensor defects list used by Sensor Defect Correction. For more information see Sensor Defect Correction support page.

Type: Enumerator.

Default value: XI_SENS_DEFFECTS_CORR_LIST_SEL_FACTORY

Usage:

```
xiSetParamInt(handle, XI_PRM_SENS_DEFECTS_CORR_LIST_SELECTOR,
XI_SENS_DEFFECTS_CORR_LIST_SEL_USER0);

xiSetParamInt(handle, XI_PRM_SENS_DEFECTS_CORR, XI_ON);
```

Value	Description
XI_SENS_DEFFECTS_CORR_LIST_SEL_FACTORY	Factory defect correction list
XI_SENS_DEFFECTS_CORR_LIST_SEL_USER0	User defect correction list
XI_SENS_DEFFECTS_CORR_LIST_SEL_IN_CAMERA	Device specific defect correction list

XI_PRM_SENS_DEFECTS_CORR_LIST_CONTENT or "sens_defects_corr_list_content"

Description: Set/Get current sensor defects list used by Sensor Defect Correction(in specific text format).

Type: String.
Default value: -
Usage:

```
xiSetParamString(handle, XI_PRM_SENS_DEFECTS_CORR_LIST_CONTENT, string, strlen(string));

xiGetParamString(handle, XI_PRM_SENS_DEFECTS_CORR_LIST_CONTENT, string, string_size);
```

XI_PRM_SENS_DEFECTS_CORR or "bpc"

Description: Correction of sensor defects. For more information see Sensor Defect Correction support page.

Type: Integer.
Default value: XI_OFF
Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_SENS_DEFECTS_CORR, &value);
xiSetParamInt(handle, XI_PRM_SENS_DEFECTS_CORR, XI_ON);
```

XI_PRM_AUTO_WB or "auto_wb"

Description: Automatic white balance.
Type: Integer.
Default value: XI_OFF
Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_AUTO_WB, &value);
xiSetParamInt(handle, XI_PRM_AUTO_WB, XI_ON);
```

XI_PRM_MANUAL_WB or "manual_wb"

Description: Manual white balance. Takes white balance from square in image center of next image received by xiGetImage. Square have 1/8th of width and height of image. The function expects white sheet of paper exposed to 50% of values (RGB values should be around 128). As result of setting of manual_wb three parameters are changed: "wb_kb", "wb_kg" and "wb_kr". User application can store them and recall when needed to set the white balance back.

Type: Integer.
Default value: 0
Usage:

```
// now camera should see the white color in approximately 50% of level
xiSetParamInt(handle, XI_PRM_MANUAL_WB, 1);
xiGetImage(handle, 1000, &image); // now API automatically calculates the white balance
```

xiGetImage(handle, 1000, &image); // this and next images will have corrected white balance

XI_PRM_WB_ROI_OFFSET_X or "wb_roi_offset_x"

Description: X offset of the area used for manual WB calculation. The sum of XI_PRM_WB_ROI_OFFSET_X and XI_PRM_WB_ROI_WIDTH must be equal or lower than the image resolution(width).

Supported cameras: MC,CB,MX

Type: Integer.

Default value: 0

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_WB_ROI_OFFSET_X, &value);
xiSetParamInt(handle, XI_PRM_WB_ROI_OFFSET_X, value);
```

XI_PRM_WB_ROI_OFFSET_Y or "wb_roi_offset_y"

Description: Y offset of the area used for manual WB calculation. The sum of XI_PRM_WB_ROI_OFFSET_Y and XI_PRM_WB_ROI_HEIGHT must be equal or lower than the image resolution(height).

Supported cameras: MC,CB,MX

Type: Integer.

Default value: 0

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_WB_ROI_OFFSET_Y, &value);
xiSetParamInt(handle, XI_PRM_WB_ROI_OFFSET_Y, value);
```

XI_PRM_WB_ROI_WIDTH or "wb_roi_width"

Description: Width of the area used for manual WB calculation. The sum of XI_PRM_WB_ROI_OFFSET_X and XI_PRM_WB_ROI_WIDTH must be equal or lower than the image resolution(width).

Supported cameras: MC,CB,MX

Type: Integer.

Default value: 0

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_WB_ROI_WIDTH, &value);
xiSetParamInt(handle, XI_PRM_WB_ROI_WIDTH, value);
```

XI_PRM_WB_ROI_HEIGHT or "wb_roi_height"

Description: Height of the area used for manual WB calculation. The sum of XI_PRM_WB_ROI_OFFSET_Y and XI_PRM_WB_ROI_HEIGHT must be equal or lower than the image resolution(height).

Supported cameras: MC,CB,MX

Type: Integer.

Default value: 0

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_WB_ROI_HEIGHT, &value);
xiSetParamInt(handle, XI_PRM_WB_ROI_HEIGHT, value);
```

XI_PRM_WB_KR or "wb_kr"

Description: White balance red coefficient.

Type: Float.

Default value: 1.0

Typical range: [0.0, 10.0]

Usage:

```
float value = 0.0;
xiGetParamFloat(handle, XI_PRM_WB_KR, &value);
xiSetParamFloat(handle, XI_PRM_WB_KR, value);
```

XI_PRM_WB_KG or "wb_kg"

Description: White balance green coefficient.

Type: Float.

Default value: 1.0

Typical range: [0.0, 10.0]

Usage:

```
float value = 0.0;
xiGetParamFloat(handle, XI_PRM_WB_KG, &value);
xiSetParamFloat(handle, XI_PRM_WB_KG, value);
```

XI_PRM_WB_KB or "wb_kb"

Description: White balance blue coefficient.

Type: Float.

Default value: 1.0

Typical range: [0.0, 10.0]

Usage:

```
float value = 0.0;
xiGetParamFloat(handle, XI_PRM_WB_KB, &value);
xiSetParamFloat(handle, XI_PRM_WB_KB, value);
```

XI_PRM_WIDTH or "width"

Description: If camera runs in single region mode this parameter represents width of the image provided by the device (in pixels). The sum of XI_PRM_OFFSET_X and XI_PRM_WIDTH must be equal or lower than the image resolution(width). Number must be divisible by the minimum increment which can be read out using the api parameter modifier XI_PRM_INFO_INCREMENT. If camera runs in multiple region mode (XI_PRM_REGION_SELECTOR) this parameter is width of region currently selected (in pixels).

Type: Integer.

Default value: Full resolution width.

Is invalidated by: XI_PRM_BINNING_HORIZONTAL, XI_PRM_DECIMATION_HORIZONTAL, XI_PRM_DOWNSAMPLING_TYPE, XI_PRM_DOWNSAMPLING, XI_PRM_IMAGE_DATA_FORMAT, XI_PRM_IMAGE_AREA, XI_PRM_OUTPUT_DATA_PACKING, XI_PRM_OUTPUT_DATA_BIT_DEPTH, XI_PRM_HEIGHT

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_WIDTH, &value);
xiSetParamInt(handle, XI_PRM_WIDTH, value);
```

XI_PRM_HEIGHT or "height"

Description: If camera runs in single region mode this parameter represents the height of the image provided by the device (in pixels). The sum of XI_PRM_OFFSET_Y and XI_PRM_HEIGHT must be equal or lower than the image resolution(height). Number must be divisible by the minimum increment which can be read out using the api parameter modifier XI_PRM_INFO_INCREMENT. If camera runs in multiple region mode () this parameter is height of region currently selected.

Note1: Changing of this parameter will remove all images from buffer queue.

Note2: In case of using small ROI in combination with Fresco FL1100 controller, please read this article.

Type: Integer.

Default value: Full resolution width.

Is invalidated by: XI_PRM_BINNING_VERTICAL, XI_PRM_DECIMATION_VERTICAL, XI_PRM_DOWNSAMPLING_TYPE, XI_PRM_DOWNSAMPLING, XI_PRM_IMAGE_DATA_FORMAT, XI_PRM_OUTPUT_DATA_PACKING, XI_PRM_IMAGE_AREA, XI_PRM_OUTPUT_DATA_BIT_DEPTH, XI_PRM_WIDTH

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_HEIGHT, &value);
xiSetParamInt(handle, XI_PRM_HEIGHT, value);
```

XI_PRM_OFFSET_X or "offsetX"

Description: Horizontal offset from the origin to the area of interest (in pixels). The sum of XI_PRM_OFFSET_X and XI_PRM_WIDTH must be equal or lower than the image resolution(width). Number must be divisible by the minimum increment which can be read out using the api parameter modifier XI_PRM_INFO_INCREMENT.

Note: Changing of this parameter will remove all images from buffer queue.

Type: Integer.

Default value: 0

Is invalidated by: XI_PRM_BINNING_HORIZONTAL, XI_PRM_DECIMATION_HORIZONTAL, XI_PRM_DOWNSAMPLING_TYPE, XI_PRM_DOWNSAMPLING, XI_PRM_IMAGE_AREA

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_OFFSET_X, &value);
xiSetParamInt(handle, XI_PRM_OFFSET_X, value);
```

XI_PRM_OFFSET_Y or "offsetY"

Description: Vertical offset from the origin to the area of interest (in pixels).The sum of XI_PRM_OFFSET_Y and XI_PRM_HEIGHT must be equal or lower than the image resolution(height). Number must be divisible by the minimum increment which can be read out using the api parameter modifier XI_PRM_INFO_INCREMENT.

Note: Changing of this parameter will remove all images from buffer queue.

Type: Integer.

Default value: 0

Is invalidated by: XI_PRM_BINNING_VERTICAL, XI_PRM_DECIMATION_VERTICAL, XI_PRM_DOWNSAMPLING_TYPE, XI_PRM_DOWNSAMPLING, XI_PRM_IMAGE_AREA

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_OFFSET_Y, &value);
xiSetParamInt(handle, XI_PRM_OFFSET_Y, value);
```

XI_PRM_REGION_SELECTOR or "region_selector"

Description: Selects Region in Multiple ROI. Parameters: XI_PRM_WIDTH, XI_PRM_HEIGHT, XI_PRM_OFFSET_X, XI_PRM_OFFSET_Y, XI_PRM_REGION_MODE are related to the particular region.

Note1: Width and offset_x could be changed only for Region 0.

Note2: Regions has to be in order from top to bottom. Region N has to start after Region N-1 ends.

Type: Integer.

Default value: 0

Is invalidated by: XI_PRM_BINNING_VERTICAL, XI_PRM_BINNING_HORIZONTAL, XI_PRM_DECIMATION_VERTICAL, XI_PRM_DECIMATION_HORIZONTAL, XI_PRM_DOWNSAMPLING_TYPE, XI_PRM_DOWNSAMPLING

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_REGION_SELECTOR, &value);
xiSetParamInt(handle, XI_PRM_REGION_SELECTOR, value);
```

XI_PRM_REGION_MODE or "region_mode"

Description: Activates/deactivates Region selected by Region Selector in Multiple ROI

Note: Region 0 is always activated, it is not possible to deactivate it.

Type: Integer.

Default value: 1 for Region selector 0 and 0 for all other regions.

Typical range: [0, 1]

Is invalidated by: XI_PRM_BINNING_VERTICAL, XI_PRM_BINNING_HORIZONTAL, XI_PRM_DECIMATION_VERTICAL, XI_PRM_DECIMATION_HORIZONTAL, XI_PRM_DOWNSAMPLING_TYPE, XI_PRM_DOWNSAMPLING

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_REGION_MODE, &value);
xiSetParamInt(handle, XI_PRM_REGION_MODE, value);
```

XI_PRM_HORIZONTAL_FLIP or "horizontal_flip"

Description: Activates horizontal flip if available in camera.

Type: Integer.

Default value: 0 for disabled flipping.

Usage:

```
xiSetParamInt(handle, XI_PRM_HORIZONTAL_FLIP, XI_ON);  //enable horizontal flipping
```

XI_PRM_VERTICAL_FLIP or "vertical_flip"

Description: Activates vertical flip if available in camera.

Type: Integer.

Default value: XI_OFF

Usage:

```
xiSetParamInt(handle, XI_PRM_VERTICAL_FLIP, XI_ON);  //enable vertical flipping
```

XI_PRM_INTERLINE_EXPOSURE_MODE or "interline_exposure_mode"

Description: Selector for Exposure parameter

Type: Enumerator.

Default value: XI_INTERLINE_EXPOSURE_MODE_OFF

Usage:

```
int interline_exposure_mode = 0;
xiGetParamInt(handle, XI_PRM_INTERLINE_EXPOSURE_MODE, &interline_exposure_mode);
xiSetParamInt(handle, XI_PRM_INTERLINE_EXPOSURE_MODE, XI_INTERLINE_EXPOSURE_MODE_OFF);
```

Value	Description
XI_INTERLINE_EXPOSURE_MODE_OFF	Disabled
XI_INTERLINE_EXPOSURE_MODE_ON	Enabled

XI_PRM_FFC or "ffc"

Description: Image flat field correction.(XI_PRM_NEW_PROCESS_CHAIN_ENABLE must be XI_ON). For more information see Flat Field Correction support page.

Type: Integer.

Default value: XI_OFF

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_FFC, &value);
xiSetParamInt(handle, XI_PRM_FFC, XI_ON);
```

XI_PRM_FFC_FLAT_FIELD_FILE_NAME or "ffc_flat_field_file_name"

Description: Set name of file of image flat field to be applied for FFC processor(in tiff format). (XI_PRM_NEW_PROCESS_CHAIN_ENABLE must be XI_ON). For more information see Flat Field Correction support page.

Note: Use the same image file for this parameter as for XI_PRM_FFC_DARK_FIELD_FILE_NAME for dark-field correction only. Processing will subtract the dark image only while using the unity (1.00) gain for correction.

Type: String.

Default value: -

Usage:

```
xiSetParamString(handle, XI_PRM_FFC_FLAT_FIELD_FILE_NAME, filename, size);
```

XI_PRM_FFC_DARK_FIELD_FILE_NAME or "ffc_dark_field_file_name"

Description: Set name of file of image dark field to be applied for FFC processor(in tiff format). (XI_PRM_NEW_PROCESS_CHAIN_ENABLE must be XI_ON). For more information see Flat Field Correction support page.

Type: String.

Default value: -

Usage:

```
xiSetParamString(handle, XI_PRM_FFC_DARK_FIELD_FILE_NAME, filename, size);
```

XI_PRM_TOF_READOUT_MODE or "tof_readout_mode"

Description: Sets ToF Readout Mode

Type: Enumerator.

Default value: XI_TOF_READOUT_MODE_A_ONLY

Usage:

```
int tof_readout_mode = 0;
xiGetParamInt(handle, XI_PRM_TOF_READOUT_MODE, &tof_readout_mode);
```

```
xiSetParamInt(handle, XI_PRM_TOF_READOUT_MODE, XI_TOF_READOUT_MODE_A_ONLY);
```

Value	Description
XI_TOF_READOUT_MODE_A_ONLY	A Only readout mode
XI_TOF_READOUT_MODE_B_ONLY	B Only readout mode
XI_TOF_READOUT_MODE_A_MINUS_B	A Minus B readout mode
XI_TOF_READOUT_MODE_A_PLUS_B	A Plus B readout mode
XI_TOF_READOUT_MODE_A_AND_B	A And B readout mode

XI_PRM_TOF_MODULATION_FREQUENCY or "tof_modulation_frequency"

Description: Sets ToF Modulation Frequency in MHz

Type: Float.

Default value: 100.0

Typical range: [4.0, 100.0]

Usage:

```
float value = 0.0;
xiGetParamFloat(handle, XI_PRM_TOF_MODULATION_FREQUENCY, &value);
xiSetParamFloat(handle, XI_PRM_TOF_MODULATION_FREQUENCY, value);
```

XI_PRM_TOF_MULTIPLE_PHASES_IN_BUFFER or "tof_multiple_phases_in_buffer"

Description: is multiple ToF phases concatenated in buffer

Type: Integer.

Default value: 0

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_TOF_MULTIPLE_PHASES_IN_BUFFER, &value);
```

XI_PRM_TOF_PHASES_COUNT or "tof_phases_count"

Description: Sets the number of tof phases. E.g. 4 for four phases.

Type: Integer.

Default value: 1

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_TOF_PHASES_COUNT, &value);
xiSetParamInt(handle, XI_PRM_TOF_PHASES_COUNT, value);
```

XI_PRM_TOF_PHASE_ANGLE or "tof_phase_angle"

Description: Sets Illumination angle for selected ToF phase

Type: Float.

Default value: 0.0

Typical range: [0.0, 360.0]

Is invalidated by: XI_PRM_TOF_PHASE_SELECTOR

Usage:

```
float value = 0.0;
xiGetParamFloat(handle, XI_PRM_TOF_PHASE_ANGLE, &value);
xiSetParamFloat(handle, XI_PRM_TOF_PHASE_ANGLE, value);
```

XI_PRM_TOF_PHASE_EXPOSURE_TIME or "tof_phase_exposure_time"

Description: Sets Exposure time for selected ToF phase in microseconds.

Type: Float.

Default value: 1000.0

Is invalidated by: XI_PRM_TOF_PHASE_SELECTOR, XI_PRM_TOF_MODULATION_FREQUENCY

Usage:

```
float value = 0.0;
xiGetParamFloat(handle, XI_PRM_TOF_PHASE_EXPOSURE_TIME, &value);
xiSetParamFloat(handle, XI_PRM_TOF_PHASE_EXPOSURE_TIME, value);
```

XI_PRM_TOF_PHASE_SELECTOR or "tof_phase_selector"

Description: Selects tof phase

Type: Integer.

Default value: 1

Usage:

```
int value = 0;
xiGetParamInt(handle, XI_PRM_TOF_PHASE_SELECTOR, &value);
xiSetParamInt(handle, XI_PRM_TOF_PHASE_SELECTOR, value);
```