

## 1. Error Code definition

MCAM_ERR_SUCCESS	0
MCAM_ERR_INVALID_BUFFER_SIZE	-1
MCAM_ERR_INVALID_HANDLE	-2
MCAM_ERR_INVALID_ID	-3
MCAM_ERR_ACCESS_DENIED	-4
MCAM_ERR_NO_DATA	-5
MCAM_ERR_ERROR	-6
MCAM_ERR_INVALID_PARAMETER	-7
MCAM_ERR_TIMEOUT	-8
MCAM_ERR_INVALID_FILENAME	-9
MCAM_ERR_INVALID_ADDRESS	-10
MCAM_ERR_FILE_IO	-11
MCAM_ERR_NOT_OPEN_DEVICE	-1000
MCAM_ERR_NO_SYSTEM	-1001
MCAM_ERR_NO_DEVICE	-1002

## 2. General functions

The return value for each function is " Error code "

```
__int32 ST_InitSystem()
```

Creates System module and initializes Library.

All functions can be used after call ST\_InitSystem()[IsInitSystem()]

```
__int32 ST_FreeSystem()
```

Quits all modules in using.

```
__int32 ST_IsInitSystem(bool* pFlag);
```

Checks that Library is available

– pFlag(IN, OUT) : IN : User value pointer, OUT : Init status(0 or 1)

```
__int32 ST_GetPortIDNum(unsigned __int32* pPortIDNum)
```

Gets the quantity of connectable Device(Camera)

- pNum(IN, OUT) : IN : User value pointer, OUT : Quantity of connectable device.

```
__int32 ST_GetPortID( unsigned __int32 portIdx, char* pPortID, unsigned __int32* pSize)
```

Gets the unique ID of selected port.

– portIdx(IN) : Port Index.

– pPortID(IN,OUT) : IN:User buffer pointer, OUT:Device ID.

– pSize(IN,OUT) : IN : User buffer size, OUT : Return string size.

```
__int32 ST_GetDeviceIDNum(unsigned __int32* pDeviceIDNum)
```

Gets the quantity of connectable devices.

- pDeviceIDNum(IN, OUT) : IN : User value pointer, OUT : quantity of connectable devices.

```
__int32 ST_GetDeviceID(unsigned __int32 deviceIdx, char* pDeviceID, unsigned __int32* pSize)
```

Gets the unique ID of selected device.

- deviceIdx(IN) : Device Index.
- pDeviceID(IN,OUT) : IN:User buffer pointer, OUT:Device ID.
- pSize(IN,OUT) : IN : User buffer size, OUT : Return string size.

```
__int32 ST_ConnectDevice(__int32* hDevice, const char* pPortID, const char* pDeviceID)
```

Connects using PortID and DeviceID's Name.

- hDevice(IN,OUT) : IN:User value pointer, OUT:Device handle.
- pPortID(IN,OUT)) : IN:Port ID, OUT: Port ID
- pDeviceID(IN,OUT)) : IN:Device ID, OUT: DeviceID

```
__int32 ST_ConnectDeviceIdx( __int32* hDevice, unsigned __int32 pPortIdx, unsigned __int32 pDeviceIdx)
```

Connects using PortID and DeviceID's Index.

- hDevice(IN,OUT) : IN:User value pointer, OUT:Device handle.
- pPortIdx(IN)) : IN:Port ID Index
- pDeviceIdx(IN)) : IN:Device ID Index

```
__int32 ST_CloseDevice(__int32 hDevice)
```

Quits the connected device.

- hDevice(IN) : Device handle.

### 3. Camera control functions

The return value for each function is " Error code "  
For Feature Name, please refer to Fig 4.

```
__int32 ST_SetIntReg (__int32 hDevice, const char* NodeName, __int32 val)
```

Sets the value in the Integer type feature.

- hDevice(IN) : Device handle.
- NodeName : Feature Name
- val(IN) : Integer Value.

```
__int32 ST_GetIntReg (__int32 hDevice, const char* NodeName, __int32* pVal)
```

Gets the value of Integer type feature.

- hDevice(IN) : Device handle.
- NodeName : NameFeature
- pVal(IN, OUT) : IN:User value pointer, OUT: Integer Value

```
__int32 ST_SetFloatReg(__int32 hDevice, const char* NodeName, double fVal)
```

Sets the value in the Float type feature.

- hDevice(IN) : Device handle.
- NodeName : Feature Name
- fVal(IN) : Float Value.

```
__int32 ST_GetFloatReg(__int32 hDevice, const char* NodeName, double *pFval)
```

Gets the value of Float type feature.

- hDevice(IN) : Device handle.
- NodeName : NameFeature
- pFval(IN, OUT) : IN:User value pointer, OUT: Float Value.

```
__int32 ST_SetBoolReg(__int32 hDevice, const char* NodeName, bool bVal)
```

Sets the value in the Boolean type feature.

- hDevice(IN) : Device handle.
- NodeName : Feature Name
- fVal(IN) : True or False.

```
__int32 ST_GetBoolReg(__int32 hDevice, const char* NodeName, bool *pBval)
```

Gets the value of Boolean type feature.

- hDevice(IN) : Device handle.
- NodeName : NameFeature
- pFval(IN, OUT) : IN:User value pointer, OUT: True or False.

```
__int32 ST_SetEnumReg(__int32 hDevice, const char* NodeName, char* val)
```

Sets the value in the Enumeration type feature.

- hDevice(IN) : Device handle.
- NodeName : Feature Name
- val(IN) : Enumeration Value. (Refer to Feature Table)

```
__int32 ST_GetEnumReg(__int32 hDevice, const char* NodeName, char* pInfo, unsigned __int32* pSize)
```

Gets the value of Enumeration type feature.

- hDevice(IN) : Device handle.
- NodeName : Feature Name
- pInfo(IN,OUT) : IN : User buffer pointer, OUT : Enumeration Value.
- pSize(IN,OUT) : IN : User buffer size, OUT : pInfo length.

```
__int32 ST_GetStrReg(__int32 hDevice, const char* NodeName, char* pInfo, unsigned __int32* pSize)
```

Gets the value of String type feature.

- hDevice(IN) : Device handle.
- NodeName : Feature Name
- pInfo(IN,OUT) : IN : User buffer pointer, OUT : String Value.
- pSize(IN,OUT) : IN : User buffer size, OUT : pInfo length.

```
__int32 ST_SetCmdReg(__int32 hDevice, const char* NodeName)
```

Sets the value in the Command type feature.

- hDevice(IN) : Device handle.
- NodeName : Feature Name

```
__int32 ST_GetIntRegRange(__int32 hDevice, const char* NodeName, __int32 *pMin, __int32 *pMax, __int32 *pInc)
```

Gets the settable range of Interger type Feature.

- hDevice(IN) : Device handle.
- NodeName : Feature Name
- pMin(IN, OUT) : IN : User value pointer, OUT : Settable minimum value.
- pMax(IN, OUT) : IN : User value pointer, OUT : Settable maximum value.
- pInc(IN, OUT) : IN : User value pointer, OUT : Settable increments.

```
__int32 ST_GetFloatRegRange(__int32 hDevice, const char* NodeName, double *pMin, double *pMax)
```

Gets the settable range of Float type Feature.

- hDevice(IN) : Device handle.
- NodeName : Feature Name
- pMin(IN, OUT) : IN : User value pointer, OUT : Settable minimum value.
- pMax(IN, OUT) : IN : User value pointer, OUT : Settable maximum value.

```
__int32 ST_GetEnumEntrySize(__int32 hDevice, const char* NodeName, __int32 *pVal)
```

Gets the quantity of Enumeration type Feature's Entry

- hDevice(IN) : Device handle.
- NodeName : Feature Name
- pVal(IN, OUT) : IN : User value pointer, OUT : Number of Entry.

```
__int32 ST_GetEnumEntryIntValue(__int32 hDevice, const char* NodeName, __int32 EntryIdx,
__int32 *pVal)
```

Gets the Integer value of Enumeration type Feature's Entry Index.

- hDevice(IN) : Device handle.
- NodeName : Feature Name
- EntryIdx : Entry Index.
- pVal(IN, OUT) : IN : User value pointer, OUT : Integer Value.

```
__int32 ST_GetEnumEntryValue(__int32 hDevice, const char* NodeName, __int32 EntryIdx, char*
pInfo, unsigned __int32 *pSize)
```

Gets the String value of Enumeration type Feature's Entry Index.

- hDevice(IN) : Device handle.
- NodeName : Feature Name
- EntryIdx : Entry Index.
- pInfo(IN,OUT) : IN : User buffer pointer, OUT : String Value.
- pSize(IN,OUT) : IN : User buffer size, OUT : pInfo length.

```
__int32 ST_GetLastError(__int32 *pErrorCode, char* pErrorDescription, unsigned __int32* pSize)
```

- pErrorCode(IN,OUT) : IN : User value pointer, OUT : Error Code.
- pErrorDescription(IN,OUT) : IN: User buffer pointer, OUT : Error Description.
- pSize(IN, OUT) : IN : User buffer size, OUT : Return string size.

## 4. Feature Information

Name	Interface	Unit	Visibility	Description
DeviceVersion	IString	R	–	Version of the device.
DeviceID	IString	R	–	This feature is deprecated (See DeviceSerialNumber).
DeviceUserID	IString	R/W	–	User-programmable device identifier.
BaudRate	IEnumeration	R/W	–	This feature controls the baud rate used by the selected serial port.
SensorTaps	IEnumeration	R/(W)	–	Number of taps of the camera sensor.
RegionSelector	IEnumeration	R/(W)	–	Selects the Region of interest to control.
RegionEnable				
Width	IInteger	R/(W)	–	Width of the image provided by the device (in pixels).
Height	IInteger	R/(W)	–	Height of the image provided by the device (in pixels).
OffsetX	IInteger	R/W	–	Horizontal offset from the origin to the region of interest (in pixels).
OffsetY	IInteger	R/W	–	Vertical offset from the origin to the region of interest (in pixels).
DecimationVertical	IInteger	R/W	–	Vertical sub-sampling of the image.
PixelFormat	IEnumeration	R/(W)	–	Format of the pixels provided by the device.
TestPattern	IEnumeration	R/W	–	Selects the type of test pattern that is generated by the device as image source.
AcquisitionFrameRate	IFloat	R/W	Hz	Controls the acquisition rate (in Hertz) at which the frames are captured.
TriggerMode	IEnumeration	R/W	–	Controls if the selected trigger is active.
TriggerSoftware	ICommand	(R)/W	–	Generates an internal trigger.
TriggerSource	IEnumeration	R/W	–	Specifies the internal signal or physical input Line to use as the trigger
TriggerDelay	IFloat	R/W	us	Specifies the delay in microseconds (us) to apply after the trigger reception before activating it.
ExposureMode	IEnumeration	R/W	–	Sets the operation mode of the Exposure (or shutter).
ExposureTime	IFloat	R/W	us	Sets the Exposure time when ExposureMode is Timed and ExposureAuto is Off.
ExposureAuto	IEnumeration	R/W	–	Sets the automatic exposure mode when ExposureMode is Timed.
AutoExposureTarget	IInteger	R/W	–	When the auto exposure is "Once" or "Continuous", the exposure time will be changed, depending on the target value.
ExposureUpdateFeature	ICommand	R/W	–	ExposureTime and ExposureAuto Update Feature.
LineSelector	IEnumeration	R/W	–	Selects the physical line (or pin) of the external device connector to configure.
LineInverter	IBoolean	R/W	–	Controls the inversion of the signal of the selected input or output Line.
LineStatus	IBoolean	R	–	Returns the current status of the selected input or output Line.
LineStatusUpdateFeature				
LineSource	IEnumeration	R/W	–	Selects which internal acquisition or I/O source signal to output on the selected Line.
UserOutputSelector	IEnumeration	R/W	–	Selects which bit of the User Output register will be set by UserOutputValue.
UserOutputValue	IBoolean	R/W	–	Sets the value of the bit selected by UserOutputSelector.
TimerDuration	IFloat	R/W	us	Sets the duration (in microseconds) of the Timer pulse.
TimerDelay	IFloat	R/W	us	Sets the duration (in microseconds) of the delay to apply at the reception of a trigger before starting the Timer.
GainRaw	IInteger	R/W	–	Controls the selected gain as an absolute physical value.
GainAuto	IEnumeration	R/W	–	Sets the automatic gain control (AGC) mode.
GainUpdateFeature	ICommand	R/W	–	GainRaw and GainAuto Update Feature
BlackLevelRaw	IInteger	R/W	–	Controls the analog black level as an absolute physical value.
BalanceRatioSelector	IEnumeration	R/W	–	Selects which Balance ratio to control.
BalanceRatio	IFloat	R/W	–	Controls ratio of the selected color component to a reference color component.
BalanceWhiteAuto	IEnumeration	R/W	–	Controls the mode for automatic white balancing between the color channels.
BalanceUpdateFeature	ICommand	R/W	–	BalanceRatio and BalanceWhiteAuto Update Feature.
LUTEnable	IBoolean	R/W	–	Activates the selected LUT.
LUTPositionSelector	IEnumeration	R/W	–	Selects the LUT Position Index.
LUTPositionX	IInteger	R/W	–	LUT Position X.
LUTPositionY	IInteger	R/W	–	LUT Position Y.
UserSetSelector	IEnumeration	R/W	–	Selects the feature User Set to load, save or configure.
UserSetLoad	ICommand	(R)/W	–	Loads the User Set specified by UserSetSelector to the device and makes it active.
UserSetSave	ICommand	(R)/W	–	Save the User Set specified by UserSetSelector to the non-volatile memory of the device.
PRNUEnable	IBoolean	R/W	–	PRNU Enable.
NoiseFilterMode	IEnumeration	R/W	–	Noise Filter Mode.
NoiseFilterLevel	IInteger	R/W	–	Noise Filter Level.
DefectFilterSelector	IEnumeration	R/W	–	Defect Filter Selector.