The Cooke Camera SDK common interface for Windows

(Uniform_SDK.dll)

Version 1.5

This manual last updated March-1-2005

The Cooke Corporation
6930 Metroplex Drive Romulus, MI 48174
Tel (248) 276-8820
Fax (248) 276-8825
info@cookecorp.com
www.cookecorp.com

Update History:
Version 1.5: SensiCam EM is added
Version 1.3: QE standard(fast-shutter) and QE double shutter are supported.
Version 1.2: An additional command STOP_CCD is added.
verus 1.1: The parameter of "timeout" for GetImage() has been redefined.

Introduction

The supplied library Uniform_SDK.dll is a software layer designed for SDKs of Cooke camera types Sensicam High Performance (including all types of QE cameras and EM) and Pixelfly. It encapsules individual SDKs specfic to each cameras to a common control interface, so that SDK programmers could apply the development of a driver for one camera type to another with minimum effort. The other purpose of this layer is to simplify camera programming. Most of the control and status monitoring are handled internally by the layer. The interface control commands presented are easy to use and self-explanatory. The interface maintains most of the features common to all the camera types, such as camera control settings and image capture.

Compared to using the the original SDKs for each camera, there are some restrictions imposed with this interface. Typically, the current interface optimizes the overall performance for sensicam, there is very little or no overhead comparing to using the original SDK; however, Pixelfly is wrapped to resemble sensicam interface at the cost of frame rate. For timing critical applications involving Pixelflys, original SDK is recommended. Users are advised to refer to original SDKs for features not included in this interface.

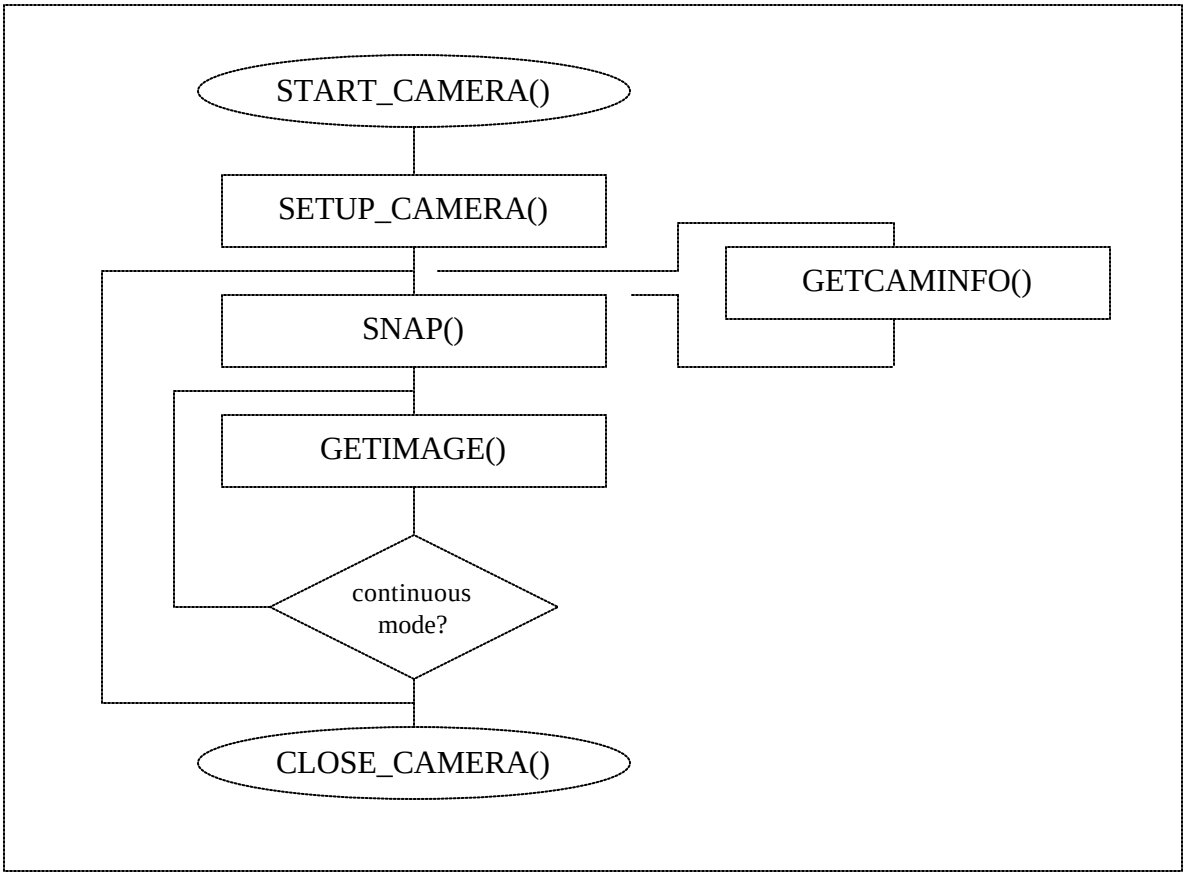
The entire interface is controlled by only 7 functions, each is described in the following sections. To build your applications, you need to load all the dlls supporting the SDK. This includes installing hardware drivers for the camera types and also placing inform_SDK.dll in the system directory (C:/Windows/System for Win98/95/ME; C:/Winnt/System32 for NT/2000). Should you load the library implicitly, Uniform_SDK.lib should be placed under your project directory or somewhere in the library search path.

When used in control of Sensicams, the funcitons are not multi-threaded safe. Each function must not be re-entried before it returns. Proper access protection is required in your application program.

An array of database structure is defined to store camera and setting information. The definition of the structure and explaination of each date field is in file <camera.h>. A unique ID is ued to identify a camera in the system and also used as entry index to the camera

information in the database. Users should not alter the contents of the ID or the database array. All the function returns are presented at the end of the manual in detail, including codes specific to each camera and those common to all.

A typical calling sequence is illustrated in the following diagram.



Control Commands

---

int SELECT_CAMERA(const char * camera_name, int board_number,
int * camIDp, CAMINFO * camdata)

This function initializes a selected camera, loads the suporting libraries for the camera, interrogates the camera, fills the camera information into the database structure, and allocates memory sufficient for one full resolution image. If successful, camdata[camID].pictptr points to the memory allocated.

This function should only be called once for each camera. If multiple cameras of the same type are used, board number 0 must be first selected.

Input:

camera_name
camera name defined as "pixelfly" or "PIXELFLY", "sensicam" or "SENSICAM".

board_number
the board selection of the camera type. Valid numbers are 0-3. If only one board exists in the system, 0 must be used

Output:

camIDp
pointer to an integer which uniquely identifies a camera connected to the system. 0-3 are dedicated to Pixelflys, 4-8 to Sensicams, and rest for future expansion.

camdata
pointer to an array of structure CAMINFO where camera and setting information is stored. The array must be defined to have at least 8 elemenets to accomondate both Sensicam and Pixefly. If successful, this function fills in basic information of the selected camera in camdata.

Special note about QE types:

All the QE cameras are also recognized as Sensicam type when controlled with this interface. At this time, there are three types of QE cameras available from Cooke: QE types QE-Long exposure (QE_L), QE fast shutter (QE_F, as known as QE standard or QE high speed), and QE double shutter (QE_D). QE_D integrates both QE_F and QE_L, that is a QE_D can be programmed to work as a QE_F or QE_L. Similarly, QE_F contains QE_L features. QE long exposure only as a stand alone camera is going obsolete but will always be supported in our software interface. SELECT_CAMERA() returns the type of your QE camera in camdata[camID].shutter. Refer to <camera.h> or GETCAMINFO() for definitions.

int SETUP_CAMERA (int camID, CAMINFO * camdata, int mode, int submode, int trigger, int roix1, int roix2, int roiy1, int roiy2, int hbin, int vbin, int gain, int delay, int exptime)

This function should be called following a successful call to SETUP_CAMERA(). If successful, it sets the operation mode of a camera, and stores the setting of camera into the CAMSETTING field of camdata. This function can also be called at any stage following SNAP(), GETIMAGE() or GETCAMINFO(). It then automatically stops the camera and resets the camera with the new parameters as specified.

Input:

camID, camdata, passed along from last return.

mode (Pixelfly/Sensicam)

0 video /long exposure,
(exposure time in ms)
1 async /QE fast shutter
(exposure time in ?s/ns)
2 double shutter
(exposure time in ms/ns)
3 EM_FAST //Sensicam EM camera only
(exposure time in ?s/ns)
4 EM_FAST_V //Sensicam EM camera only
(exposure time in ?s/ns)

submode (Sensicam when only), specifes readout order and board output when operating in long exposure mode. When in other modes, the output is fixed to busy-out.

submode	0	sequential, busy out
	1	simultaneous, busy out
	2	sequential, expos out*
	3	simultaneous, expos out*

It is recommended that “simultaneous” submode setup should not be used in combination with single snap mode (refer to SNAP command for running modes)

trig	0	software trigger
	1	hardware trigger rising edge
	2	hardware trigger falling edge

roix1, roix2 , roiy1, roiy2 define boundaries for ROI in 32 pixel units with a possible exception of the last section for special cameras. The possible values are determined by camera resolution. For Sensicams, this setting affects the readout of the CCD-Chip. Less data is transferred, but readout time is not changed; while for Pixelflys, Pixelfly hardware does not include ROI capability, the functionality presented in this context is constructed purely through software. Hence using ROI on Pixelfly does not have any gain, instead a minor performance loss is suffered.

roix1	start value of horizontal ROI	
roix2	end value of horizontal ROI	
	range 1 ... 20	for CCD chip type 640 x 480
	range 1 ... 40	for CCD chip type 1280 x 1024
	range 1 ... 43	for CCD chip type 1376 x 1040 or 1360x1024
	range 1 ... 32	for EM 1024x1002

roiy1	start value of vertical ROI	
roiy2	end value of vertical ROI	
	range 1 ... 15	for CCD chip type 640 x 480
	range 1 ... 32	for CCD chip type 1280 x 1024
	range 1 ... 33	for CCD chip type 1376 x 1040
	range 1 ... 32	for EM 1024x1002

Normally the smallest ROI is 32 pixels in square. Please also notice the special resolutions which are not divisibale by 32. The last section of ROI would be remaining pixels in such instances. To get for example the upper right corner 32*32 pixel the ROI settings should be roix1=1, roix2=1, roiy1=1, roiy2=1.

hbin sets the horizontal binning. This setting affects the readout of the CCD-Chip. Less data is transferred but the readout time is not changed.

hbin	horizontal binning 1 no binning other possible values 2 for Pixelfly and 2, 4, 8 for Sensicam

vbin sets the vertical binning. The maximal vertical binning setting depends on the selected vertical ROI. This setting affects the readout of the CCD-Chip. Less data is transferred and the readout time is decreased.

vbin	vertical binning 1 no binning other possible values: Pixelfly: 2 Sensicam SVGA 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 Sensicam VGA 2, 4, 8, 15, 16, 30, 32, 60, 64, 120, 128, 240, 256, 480 QEs 2, 4, 8, 16 EM 2, 4, 8, 16, 32

Note: ROI procedes binnings in controlling image dimensions. ROI is defined on 1x1 image and binnings are applied to the ROI. For example, a setting of [roix1, roix2, roiy1, rioy2, hbin, vbin] = [1, 4, 1, 5, 2, 4] on the resolution of 1280x1024 would yield an image of 64x40.
For EM camera, bining can only apply to the ROIs what have full 32x32 pixels, roiy2 should be 31 or less.

gain	0 normal analog gain 1 extended analog gain 3 Low Light Mode (only for QE types) SensiCam EM exclusive: 2, 5, 10, 20, 50, 100, 200, 500, 1000

delay	Defined for Sensicam only, 0 to 1000000 ms for Sensicam and QE_L 0 to 1000000 ns in steps of 200ns, for QE fast shutter and double shutter 0 to 75000 µs, in steps of 75 µs. for EM_FAST *delay must be set 0 for simultaneous submode and EM_FAST_V

exptime	exposure time Pixelfly single async mode 10…10,000µs video mode 1…10,000ms Sensicam ,EM and QE_L 1 to 1,000,000ms QE fast shutter 500 to 10,000,000 ns in steps of 200ns QE double shutter not applicable. EM_FAST, EM_FAST_V 75 to 15000 µs, in steps of 75 µs.
For QE_D, the exposure times of the two images are determined by the sequence of the TRIG input signal on the PCI Interface board. No times values can be set through software	

Output: none

int SNAP (int camID, CAMINFO * camdata, int mode)

This command must be called following a successful call to SETUP_CAMERA(). It triggers the camera for a single exposure or continuously capturing. In single capture mode, this function must be called for each exposure; while in continuous mode, this command is called only once, the camera is triggered automatically whenever image buffer is available for recording image. For PixelFly cameras, the input "mode" is a dummy variable. ASYNC setup could only capture single images, each snap only yields one image; while VIDEO setup produces continuous flow of images once SNAP() is issued and provided that an image buffer is available. To start capture, this command must be called for both cameras

Input:

camID, camdata,	as passed along
mode	Sensicam 0 single capture 1 continuous

Output: none

int GETIMAGE (int camID, CAMINFO * camdata, DWORD timeout)

This command must be called following a successful call to SNAP(). It polls the board for image transfer status. Once available it loads the image into the previously allocated user space. It also prepares the image buffer for next exposure. In the case of continuous capturing mode, looping GETIMAGE() itself would acquire a series of images.

"timeout" can be specified by the user to coordinate with special hardware. The function exits unconditionally after timeout expires. The recommended value is exposure time as the minimum.

Input:

camID	camdata defined as previous
timeout	the maximum waiting time tolerated for the exposure

Output: none

int STOP_CCD (int camID, CAMINFO * camdata)

A successful call to this additional command stops a running exposure and clears CCD. Any unfinished transfer of images gets lost. The camera setting from last SETUP_CAMERA() remains in effect. This command can be used following GETIMAGE() and followed by SNAP(), SETUP_CAMERA(), GETCAMINFO() or CLOSE_CAMERA().

Input:

camID, camdata,	as passed along

Output: none

int GETCAMINFO (int camID, CAMINFO * camdata, int *color, int *shutter, int * x, int * y, int * ele_temperature, int * ccd_temperature)

This function can be called at any stage after SELECT_CAMERA() and before CLOSE_CAMERA(), it outputs the camera type information individually and reports the temperatures of internal electronics and ccd.

Input:

camID andcamdata defined as previous

Output:

color	nonzero indicates a colored CCD
shutter	nonzero indicates double shutter for pixefly 1 for fast shutter, 2 for double shutter for QEs
x, y	the resolution of the camera
ele_temp	temperature of internal electronics
ccd_temp	temperature of ccd

int CLOSE_CAMERA (int camID, CAMINFO * camdata)

This function should only be calledfor each camera. It stops the camera and release all the resources associated with using the camera.

After this command is used, SELECT_CAMERA() must be called again to reuse the camera.

Input:

CamID, camdata	as passed along from previous operations

Output: none

Sample code:

```
#include “camera.h”
…
 int camID;
CAMINFO camdata[8];
int boardnum;
……

{
...
err=SELECT_CAMERA(“sensicam”, 0, &camID, camdata);
err=SETUP_CAMERA(camID,camdata,0, 0, 0,1,20 , 1, 20,1, 1,Gain,10,30);
err=SNAP(camID,camdata,0);
err=GETIMAGE(camID,camdata, 2000);
//the image is loaded into memory pointed by camdata[camID].pictptr
```

…….
CLOSE_CAMERA(camID,camdata);
…..
}

Return Codes

The following is a list of return codes defined for the interface. The error codes are designed for debugging on different levels, some may not appear useful for SDK users.

Function ok

0	no error, function call successful

Common Error

-50	function not defined in supporting dll
-51	board number is invalid
-52	camera type not recognised
-53	memory allocation failed
-54	timeout in reading image
-55	ROI or binnings inputs not valid
-56	supporting dll missing
-58	no driver found for this OS platform
-59	calling sequence error, not initialized

Sensicam specific errors

-1	initialization failed; no camera connected
-2	timeout in any function
-3	function call with wrong parameter
-4	cannot locate PCI card or card driver
-5	cannot allocate DMA buffer
-6	reserved
-7	DMA timeout
-8	invalid camera mode
-9	no driver installed
-10	no PCI bios found
-11	device is hold by another process
-12	error in reading or writing data to board
-13	wrong driver function
-14 ...-19	reserved
-20	LOAD_COC error (camera runs program memory)
-21	too many values in COC
-22	CCD temperature or electronics temperature out of range
-23	buffer allocate error
-24	READ_IMAGE error
-25	set/reset buffer flags is failed
-26	buffer is used
-27	call to a Macintosh function is failed
-28	DMA error

-29	cannot open file
-30	registry error
-31	open dialog error
-32	needs newer called vxd or dll

Warnings

100	no image in PCI buffer
101	picture too dark
102	picture too bright
103	one or more values changed
104	buffer for builded string too short

PixelFly Specific Errors

-101	initialization failed; no camera connected
-102	timeout in any function
-103	function call with wrong parameter
-104	cannot locate PCI card or card driver
-105	wrong operating system
-106	no or wrong driver installed
-107	IO function failed
-108	reserved
-109	invalid camera mode
-110	reserved
-111	device is hold by another process
-112	error in reading or writing data to board
-113	wrong driver function

-201	timeout in any driver function
-202	board is hold by an other process
-203	wrong boardtype
-204	cannot match processhandle to a board
-205	failed to init PCI
-206	no board found
-207	read configuration registers failed
-208	board has wrong configuration
-210	camera is busy
-211	board is not idle
-212	wrong parameter sent
-213	head is disconnected
-216	board initialization FPGA failed
-217	board initialization NVRAM failed
-221	not enough IO-buffer space for return values
-230	picture buffer not prepared for transfer
-231	picture buffer in use
-232	picture buffer hold by another process
-233	picture buffer not found

-234	picture buffer cannot be freed
-235	cannot allocate more picture buffer
-236	no memory left for picture buffer
-237	memory reserve failed
-238	memory commit failed
-239	allocate internal memory LUT failed
-240	allocate internal memory PAGETAB failed
-248	event not available
-249	delete event failed
-256	enable interrupts failed
-257	disable interrupts failed
-258	no interrupt connected to the board
-264	timeout in DMA
-265	no dma buffer found
-266	locking of pages failed
-267	unlocking of pages failed
-268	DMA buffersize to small
-269	PCI-Bus error in DMA
-270	DMA is running, command not allowed
-328	get processor failed
-329	reserved
-330	wrong processor found
-331	wrong processor size
-332	wrong processor device
-333	read flash failed

Customer Service

Should you encounter a problem or a question about matters not handled in these operating instructions, we recommend contacting us:

... by phone **(248) 276-8820**
... by fax **(248) 276-8825**
... by email service@cookecorp.com

For a quicker reply we need following information:

- **Serial number of the camera head**
- **Serial number of the PCI-Controller-Board**
- Detailed description of the problem
- Experiment conditions
- Settings in the camera control software
- Technical data of the computer