e-CAMView

e-CAMView Extented Library API Document

Revision 1.2
May-17-2012

Contents

1 Revision History

Rev No	Date	Major Changes	Author
1.0	May 7, 2012	Initial Draft specifications for e-CAMView extended library	Camera Product Team
1.1	May 10,2012	Added more description	Camera Product Team

2 Introduction

The **e-CAMView** is a DirectShow based camera application to be used with the USB Camera products from XtendLan. Some of the USB Camera products offered by XtendLan have an UVC Extension Unit (UVC XU) interface that allows the users and developers to implement some of the features that are not part of the standard USB UVC interface. In order to make use of these features that are supported by UVC XU, XtendLan has developed an USB Extended Library that encapsulates these UVC XU commands and exports to the developer as simple API set. The developers can use these APIs in their customized DirectShow camera application to implement the special features offered by XtendLan's USB cameras. The demo application e-CAMView that is provided by XtendLan has been integrated with this Extended Library. This document describes about these APIs that are supported by XtendLan's Extended Library.

3 Scope of this Document

The e-CAMView application is built on top Microsoft DirectShow framework using the DirectShow APIs. There is an extensive documentation provided by Microsoft for using the DirectShow frame and reference manual for DirectShow APIs. These APIs in general support the standard USB camera controls and the developers must refer to MSDN documentation for anything related to DirectShow APIs. The MSDN documentation is an extensive source of information for DirectShow framework, APIs and references, including the sample code snippets.

However, as explained above, some of the UVC XU features and controls are supported by the USB Camera module developer XtendLan. This document explains only the about these Extended Library APIs which are used to control the additional features supported by the XtendLan USB cameras. Additional features that are supported by the XtendLan USB cameras are

- Autofocus Trigger
- Manual Focus
- General Purpose output pin control (for controlling illumination LEDs etc)
- Special purpose and general input pins that are specific to each USB cameras

The developers are strongly recommended to use this document along with the product documentation of respective CARDUSB camera modules that they are using. These APIs are usually generic and specific differences between each of CARDUSB cameras can be found only by referring the corresponding product documentation.

4 Prerequisites

It is assumed that the developer is very familiar with the Microsoft DirectShow framework and also the DirectShow APIs and terminology. For any more information on DirectShow, the developer is recommended to visit MSDN library for more documentation.

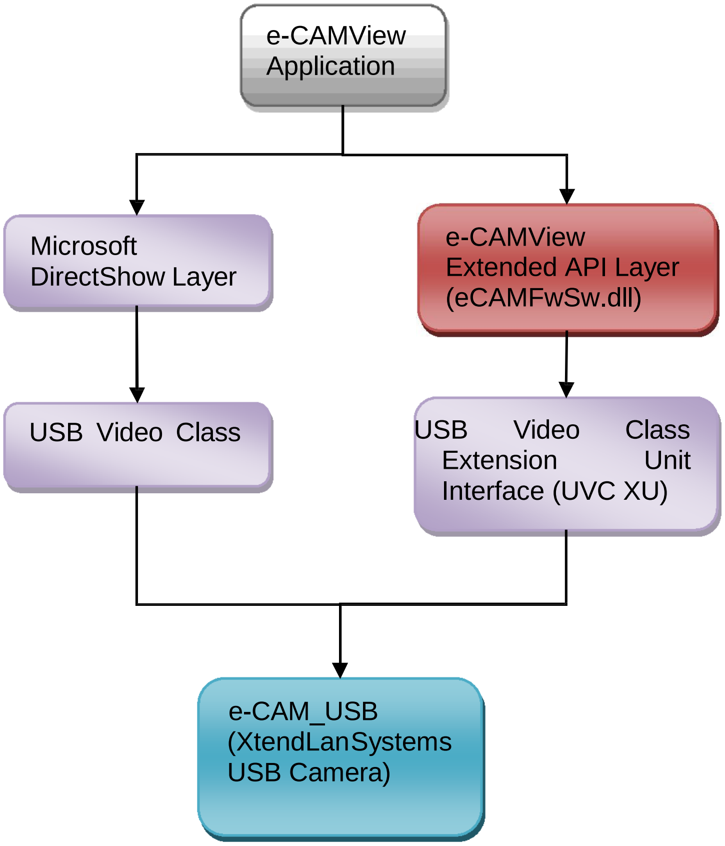
In order to use the Extended Library APIs, the developer should have registered the UVC XU filter in their machine. Usually this is installed when installing our e-CAMView application on the PC.

It is also assumed that the developer is familiar with C and C++ programming and camera or more specifically USB Camera modules.

The developer has also installed the necessary DirectShow application development tools and the DirectShow SDK and the developer is familiar with the DirectShow application development process.

5 Block Diagram

Following is the software block diagram of the e-CAMView camera application working. As it can be seen, the e-CAMView application is using the DirectShow interface and also the UVC XU interface through the Extended Library. In order to use the Extended library APIs, the developer should first install UVC XU DLL in their PC/platform and then integrate their application using our Extended Linrary (e-CAMFWSw.dll). Only then the XU interface will work.


e-CAMView Application
Microsoft DirectShow Layer
e-CAMView Extended API Layer (eCAMFwSw.dll)
USB Video Class
USB Video Class Extension Unit Interface (UVC XU)
e-CAM_USB (XtendLanSystems USB Camera)

All the UVC XU features are encapsulated in the e-CAMView Extended API layer as shown in the figure. There is no mandatory requirement to use the Extended Library in the customer applications. The e-CAMView application shall run without being linked against the Extended Library or in deed can work with any standard DirectShow cameras with standard DirectShow features.

As mentioned earlier, Extended API usage is not required for normal use cases where the customers are interested in preview of video in various resolutions and image capture requirements. In such cases, customers can develop applications using only the standard DirectShow APIs without using any of the Extended APIs. The XtendLan supplied Extended API library is required only in special cases.

6 APIs Description

All these Extended APIs require that the camera is functional and streaming the video. The Extended API calls will fail, while the camera is not streaming video.

6.1 InitLibrary

Function Prototype:
INT32 InitLibrary(UINT8 *DeviceName)

Return Value:

1€Successful.
-1(0xFFFF FFFF)€Failed.

Description:

DeviceName €A unique string that identifies the device

This API should be called with the DeviceName of the USB Camera with which the Vendor Specific Operations will be done.

To retrieve the DeviceName, please refer this MSDN link (or e-CAMView sample code) to enumerate the connected video input devices to the PC.

http://msdn.microsoft.com/en-us/library/windows/desktop/dd377566%28v=vs.85%29.aspx

From the above link documentation, retrieve and pass on the "Devicepath" string in place of "DeviceName" for this API.

If the passed devicename to this API belongs to the XtendLan product, then the required UVC XU (Extension Unit) controls are registered with the uvcvideo driver and the UVC XU control window will be available in the e-CAMView Application.

If the passed device name to this API is not belonged to the XtendLan product, then the UVC XU control for this device will not be available. Also the API will return -1.

This API must be called before invoking any other Extension Library APIs.If this APIs is not called, then the other APIs will just return the FAILURE (i.e.,-1 or 0xFFFF FFFF) code.

6.2 DeInitLibrary

Function Prototype:
INT32 DeInitLibrary(UINT8 *DeviceName)

Return Value:
1€Successful.
-1(0xFFFF FFFF)€Failed.

Description:
If the device is an econ product, then the UVC XU controls are de-registered and removed in the e-CAMView Application.

This API must be called at the time of closing the DirectShow application in which Extension Library is used.

6.3 AFFreezeControl

Function Prototype:

INT32 AFFreezeControl(INT32 FreezeControl)

Return Value:

1€Successful.
-1(0xFFFF FFFF)€Failed.

Description:

The AFFreezeControl API will be useful only when the current focus mode is Auto Focus. The FreezeControl will be one of the followings:

1) MANUAL FOCUS – 0x03(This option will switch the camera to manual focus control mode). This mode is used to set the focus position manually.
2) AF_SINGLE_TRIGGER – 0x05(This option will trigger the autofocus function. This is an one-shot AF trigger. The camera adjusts itself to its best focus position). The AF position remains unchanged even if there is any change in the scene. If there is any change in the scene or object, we have to give this command once again to trigger autofocus.

6.4 GetAFStatus

Function Prototype:

INT32 GetAFStatus(INT32 *AFStatus)

Return Value:

1€Successful.
-1(0xFFFF FFFF)€Failed.

Description:

This API fills the current AF Status in the memory location pointed by the AFStatus pointer variable. On successful return of this API, the AFStatus value can be interpreted as

AfStatus	**Description**
0x1,0x2	Auto Focus Mode
0x3	Manual Focus Mode
0x5	AF is Single Trigger Focusing
0x6	AF is Single Trigger Focused

6.5 ControlGPO

Function Prototype:

INT32 ControlGPO(INT32 GPIONumber, INT32 Value)

Return Value:

1€Successful.
-1(0xFFFF FFFF)€Failed.

Description:

This API is used to set the required state for the General Purpose Output Pin (GPO). Currently the CARDUSB cameras have single GPO (GPIO8). This GPO pin is denoted as DO0 in the camera module datasheet. The following section describes the mapping of pins mentioned in the camera module datasheet with this API reference.

The “GPIONumber” should be 8
And the “Value” should be either 0(Clear or Low State) or 1(Set or High State)

The current state of this GPO cannot be known by this API. This API can be used only to Set or Clear the GPO pin state.

6.6 GetGPIStatus

Function Prototype:

INT32 GetGPIStatus(INT32 *GPIStatus)

Return Value:

1€Successful.
-1(0xFFFF FFFF)€Failed.

Description:

This API is used to get the status of the all three General Purpose Inputs (GPIO3, GPIO6 and GPIO7) in CARDUSB camera.

The 0th bit in the 32 bit value pointed by the GPIStatus indicates the state of GPIO7
The 1st bit in the 32 bit value pointed by the GPIStatus indicates the state of GPIO6
The 2st bit in the 32 bit value pointed by the GPIStatus indicates the state of GPIO3
(0€Low State,1€High State)

6.6.1 Pin Mapping

The APIs denote the general purpose I/O pins as GPIO pins. However, the camera module datasheet refer the general purpose I/O pins with a different naming convention. This table describes the mapping of the general purpose I/O pins.

Pin Name as per e-CAM51_USB Datasheet	Pin Name as per e-CAM33_USB Datasheet	GPIO Number as referred in this API manual	Remarks
TRIG	TRIG	GPIO6	
DI0	DI0	GPIO7	
DO0	DO0	GPIO8	
F/M	F/M	GPIO3	

6.7 GetSensorId

Function Prototype:

INT32 GetSensorId(INT32 *SensorId)

Return Value:

1€Successful.
-1(0xFFFF FFFF)€Failed.

Description:

This API is used to get the Sensor ID of the camera module in the CARDUSB camera.

The return value expected for each camera module is given below:

USB Camera	Expected Return Value
CARDUSB3	0x364c
CARDUSB5	0x5640

6.8 GetFirmwareVersionInfo

Function Prototype:

INT32 GetFirmwareVersionInfo(TCHAR *FirmwareVersion)

Return Value:

1€Successful.
-1(0xFFFF FFFF)€Failed.

Description:

This API is used to get the version of the firmware running in the CARDUSB camera. Make sure pass the TCHAR array of size 255 to get the firmware version info details.

Please mention this Firmware version string in all your support related queries.

6.9 SetManualFocusPosition

Function Prototype:

INT32 SetManualFocusPosition(INT16 FocusPosition)

Return Value:

1€Successful.
-1(0xFFFF FFFF)€Failed.

Description:

This API is used to change the manual focus position. This API can be used only when the current focus mode is Manual Focus Mode.
The “FocusPosition” value should be between **0** and **255.**If the value goes beyond this range then the API will return the failure code.

0th Postion€Far Object
255th Postion€ Near Object.

6.10 GetManualFocusPosition

Function Prototype:

INT32 GetManualFocusPosition(INT16 *FocusPosition)

Return Value:

1€Successful.
-1(0xFFFF FFFF)€Failed.

Description:

This API is used to get the current focus lens position. This API can be used in any focus mode either Auto focus or in the manual mode. The return value is an unsigned 8-bit value (0 to 255) indicating the current lens position.

7 List of Reserved APIs

The followings are the list of the APIs used for doing specific operation with the CARDUSB camera. These APIs are reserved and not recommended to use in the applications developed by non-econ engineers.

7.1 GetUVCChipId

Function Prototype:

INT32 GetUVCChipId(INT16 *UVCChipId)

Return Value:

1€Successful.

-1(0xFFFF FFFF)€Failed.

Description:

This API is used to get the UVC Chip ID from the EEPROM memory of the CARDUSB camera.

8 Conclusion

This document will help the developers to integrate the extended features of the XtendLan USB cameras into their application.