**Implementing Still Capture with Hardware Trigger:**

**Objective:**

This document describes the implementation of still image capture in a UVC device by hardware trigger.

**Brief Description of the Implementation:**

This project was implemented using superspeed explorer kit and the image sensor MT9M114. The host applications e-CamView and AMCap were used for testing this project. In this project, FX3 streams 720p video data and captures a frame of same resolution when switch SW2 is pressed in USB 3.0 mode. In USB 2.0 mode, FX3 streams video data of VGA resolution and captures a frame of same resolution when switch SW2 is pressed.

**Overview:**

A common feature of video cameras is the support of still image capture associated with a video stream. Still images can be captured by software trigger or hardware trigger. There are 3 methods for capturing still images from a video stream that are supported by UVC 1.1 spec. UVC device will have to specify which method it supports in the relevant descriptors. The 3 methods for capturing still images from a video stream which are supported by UVC 1.1 spec are:

**Method 1:** The host software will extract the next available video frame from the active video pipe in the relevant VideoStreaming interface upon receiving the hardware trigger event. The hardware does not interrupt or alter the video stream in this case. For this method, the still image frame is always the same size as the video frames being streamed

**Method 2:** If the device supports higher-quality still images, it has the option of streaming still image-specific packets across the active video pipe. In this case, the host software will temporarily suspend video streaming, select the optimal bandwidth alternate setting based on the still probe/commit negotiation (subject to bandwidth availability), send a VS\_STILL\_IMAGE\_TRIGGER\_CONTROL Set request with the "Transmit still image" option, and prepare to receive the still image data. The device transmits the still image data marked as such in the payload header. Once the complete still image is received, the host software will then revert back to the original alternate setting, and resume video streaming.

**Method 3:** This method enables the capture of higher-quality still images from a dedicated bulk still image pipe. By doing so, the active streams would continue uninterrupted.

**Modifications on project associated with AN75779:**

The modifications that are to be done on the project associated with AN75779 to implement method 2 of still capture are described below:

**1. In cyfxuvcdscr.c file:**

a. Inside **Class-specific Video Streaming Input Header Descriptor**, which can be found inside CyFxUSBHSConfigDscr[ ] and CyFxUSBSSConfigDscr[ ], the following changes are to be made.

* wTotalLength : 0x51,0x00
* bStillCaptureMethod : 0x02
* bTriggerSupport : 0x01

In order to understand why these fields are set as described above, refer to Table 3-13 of [UVC 1.1](http://www.cajunbot.com/wiki/images/8/85/USB_Video_Class_1.1.pdf) spec.

The snapshot below shows the modified Class-specific Video Streaming Input Header Descriptor inside CyFxUSBSSConfigDscr[ ].



b. Within the **Class-specific Uncompressed VS Frame descriptor** of CyFxUSBHSConfigDscr[ ] and CyFxUSBSSConfigDscr[ ], modify the **bmCapabilities** field to 0x02.

In order to understand why this field is set as described above, refer to Table 3-2 found in [UVC 1.1 spec for Uncompressed Payload](http://www.cajunbot.com/wiki/images/8/88/USB_Video_Payload_Uncompressed_1.1.pdf).

The snapshot below shows the modified Class specific Uncompressed VS frame descriptor inside CyFxUSBSSConfigDscr[ ].



c. After the **Class-specific Uncompressed VS Frame descriptor** of CyFxUSBHSConfigDscr[ ] and CyFxUSBSSConfigDscr[ ], add a **Still Image Frame Descriptor**. The Still Image Frame Descriptor of CyFxUSBSSConfigDscr[ ] for capturing a frame of resolution 1280\*720 pixels is shown below:



To understand how to write this descriptor, refer to table 3-17 of the [UVC 1.1](http://www.cajunbot.com/wiki/images/8/85/USB_Video_Class_1.1.pdf) Spec

d. Now, update the **Length of this descriptor and all sub descriptors field of CyFxUSBHSConfigDscr[ ] and CyFxUSBSSConfigDscr [ ]** to 0xD8,0x00 and 0xE4,0x00 respectively.

**2. In uvc.c file:**

**a.** Include a ‘Video Still Probe Control’ structure for USB 3.0 as well as USB 2.0 as shown below:

uint8\_t glStillProbeCtrl30[CY\_FX\_MAX\_STILL\_PROBE\_SETTING] ={

 0x01, /\*bFormatIndex: Use 1st Still Format Index\*/

 0x01, /\*bFrameIndex: Use 1st Still Frame Index\*/

 0x00, /\*bCompressionIndex: No Compression supported\*/

 0x00,0x20,0x1C,0x00, /\*dwMaxVideoFrameSize: 1280\*720\*2 bytes\*/

 0x00,0x40,0x00,0x00 /\*dwMaxPayloadTransferSize:16KB DMA buffer size\*/

};

uint8\_t glStillProbeCtrl20[CY\_FX\_MAX\_STILL\_PROBE\_SETTING] ={

0x01, /\*bFormatIndex: Use 1st Still Format Index\*/

0x01, /\*bFrameIndex: Use 1st Still Frame Index\*/

0x00, /\*bCompressionIndex: No Compression supported\*/

0x00, 0x60, 0x09, 0x00, /\*dwMaxVideoFrameSize: 640\*480\*2 bytes\*/

0x00, 0x40, 0x00, 0x00 /\*dwMaxPayloadTransferSize:16KB DMA buffer size\*/

};

Refer to Table 4-50 of the [UVC 1.1](http://www.cajunbot.com/wiki/images/8/85/USB_Video_Class_1.1.pdf) spec to understand about the different fields in Video Still probe and commit control structure.

**b.** Handle VS\_STILL\_PROBE\_CONTROL, VS\_STILL\_COMMIT\_CONTROL and VS\_STILL\_IMAGE\_TRIGGER\_CONTROL inside UVCHandleVideoStreamingRqts() function. Refer to Table 4-51, 4-52 and 4-54 of [UVC 1.1](http://www.cajunbot.com/wiki/images/8/85/USB_Video_Class_1.1.pdf) Spec to understand about VS\_STILL\_PROBE\_CONTROL Requests, VS\_STILL\_COMMIT\_CONTROL Requests and Still Image Trigger Control

**c.** In order to make a hardware trigger for still capture, GPIO 45 is used. A callback function is registered during the initialization of the GPIO block. This part of the code is written inside **CyFxUVCApplnInit()** function and is shown below:

apiRetStatus = CyU3PGpioInit (&gpioClock, CyFxGpioIntrCb);

Now, GPIO 45 is overridden using the API CyU3PDeviceGpioOverride() and then configured as input with interrupt mode as CY\_U3P\_GPIO\_INTR\_NEG\_EDGE.

This part of the code is shown below:



Now, inside the callback function, the state of GPIO 45 is read. In the explorer kit, switch SW 2 is connected to GPIO 45. The connection is made in such a way that when the switch is released, GPIO 45 is pulled up. When the switch is pressed, GPIO 45 goes low.

**d.** Create a DMA channel from CPU to Status Interrupt Endpoint. This section of code is shown below:

dmaConfig.size = 16;

dmaConfig.count = 0;

dmaConfig.prodSckId = (CyU3PDmaSocketId\_t) *CY\_U3P\_CPU\_SOCKET\_PROD*;

dmaConfig.consSckId = (CyU3PDmaSocketId\_t) (*CY\_U3P\_UIB\_SOCKET\_CONS\_0* | CY\_FX\_EP\_CONTROL\_STATUS\_SOCKET);

dmaConfig.prodAvailCount = 0;

dmaConfig.prodHeader = 0;

dmaConfig.prodFooter = 0;

dmaConfig.consHeader = 0;

dmaConfig.dmaMode = *CY\_U3P\_DMA\_MODE\_BYTE*;

dmaConfig.notification = 0;

dmaConfig.cb = 0;

apiRetStatus = **CyU3PDmaChannelCreate**(&Status\_Interrupt\_Channel,*CY\_U3P\_DMA\_TYPE\_MANUAL\_OUT*, &dmaConfig);

**if**(apiRetStatus != *CY\_U3P\_SUCCESS*)

 {

 **CyU3PDebugPrint**(2, "Status Channel creation failed with 0x%x\n", apiRetStatus);

 CyFxAppErrorHandler(apiRetStatus);

 }

**Logical flow of the implementation of still image capture using Hardware trigger:**

1. Once the video streaming application is started, press the switch SW2.

2. This will trigger a GPIO interrupt and the callback function (CyFxGpioIntrCb) will be invoked. Inside the callback function, the state of the GPIO is read and if the state is low, then a flag Button\_Press\_Event will be set.

3. Inside the infinite loop of UVCAppThread\_Entry (), check whether the flag Button\_Press\_Event is set or not. If the flag Button\_Press\_Event is set, then clear the flag and call the function Send\_Status\_to\_Host ().

4. Inside the Send\_Status\_to\_Host () function, fill the 4 bytes status interrupt data into DMA buffer and send it over the CPU->Status Interrupt Endpoint DMA channel to Host. Refer to Table 2-1 and 2-3 of the [UVC 1.1](http://www.cajunbot.com/wiki/images/8/85/USB_Video_Class_1.1.pdf) Spec to understand about the 4 byte status interrupt data that is to be send to the host.

5. This Status Packet informs the Host Application about a button press event. Since, in the UVC descriptor we have informed the Host Application to initiate Still Capture on receiving a hardware trigger interrupt event from the device (bTriggerUsage:0x00), the Host App will send a VS\_STILL\_IMAGE\_TRIGGER\_CONTROL with bTrigger set to 1.

6. When the device receives a VS\_STILL\_IMAGE\_TRIGGER\_CONTROL with bTrigger field set to 1, a flag ‘Still\_Flag’ will be set.

7. Since, button press event is an asynchronous one, at the time FX3 receives a VS\_STILL\_IMAGE\_TRIGGER\_CONTROL from the Host, it might be half-way through the streaming of a particular frame. So, FX3 has to send a frame with Still Image Header once the current frame is completely transferred. CyFxUVCAddHeader () function has been modified to implement this procedure. Refer to Table 2-5 of [UVC 1.1](http://www.cajunbot.com/wiki/images/8/85/USB_Video_Class_1.1.pdf) Spec to understand about the format of payload header.