Advanced Micro Devices

Advanced Media Framework –Video Converter

Programming Guide

Disclaimer

Copyright Notice

MIT license

Contents

1 Introduction

This document provides a complete description of the AMD Advanced Media Framework (AMF) Video Converter Component. This component performs the following functions:

- Color space conversion
- Color format conversion
- Gamma correction
- Scaling

2 AMF Video Converter Component

Video Converter accepts input frames stored in *AMFSurface* objects wrapping DirectX 9 surfaces, DirectX 11 textures, OpenGL or OpenCL surfaces. The output is placed in *AMFSurface* objects wrapping DirectX 9 surfaces, DirectX 11 textures, OpenGL or OpenCL surfaces, depending on the component configuration.

Include *public/include/components/VideoConverter.h*

2.1 Component Initialization

The AMF Video Converter component should be initialized using the following sequence:

1. Create an AMF Context and initialize it for one of the following:
    a. DirectX 11.1
    b. DirectX 9
    c. OpenGL
    d. OpenCL
2. Configure the Converter component by setting the necessary properties using the *AMFPropertyStorage::SetProperty* method on the converter object.
3. Call the *AMFComponent::Init* method of the converter object.

2.2 Configuring the Converter

The *format, width* and *height* parameters of the *AMFComponent::Init* method describe the input stream. Parameters of the output stream are set using the following properties:

- *AMF_VIDEO_CONVERTER_OUTPUT_FORMAT* – specifies the output color format/space. Can be one of the following values:
    - *AMF_SURFACE_NV12* – convert to NV12
    - *AMF_SURFACE_BGRA* – convert to BGRA
    - *AMF_SURFACE_YUV420P* – convert to YUV 4:2:0 (progressive only)
- *AMF_VIDEO_CONVERTER_MEMORY_TYPE* – specifies the memory type of output surfaces (surfaces are allocated internally by the Converter component). Can be one of the following values:
    - *AMF_MEMORY_DX11* – place output in a DirectX 11 texture
    - *AMF_MEMORY_DX9* – place output in a DirectX 9 surface
    - *AMF_MEMORY_UNKNOWN* – retain the same memory type as input (no interop)
- *AMF_VIDEO_CONVERTER_OUTPUT_SIZE* – output image resolution specified as *AMFSize*. Scaling will be performed when this property is set.
- *AMF_VIDEO_CONVERTER_OUTPUT_RECT* – specifies the target rectangle in the output surface to scale the image into as *AMFRect*.
- *AMF_VIDEO_CONVERTER_KEEP_ASPECT_RATIO* – force the scaler to keep the aspect ratio of the input image when the output size specified by the *AMF_VIDEO_CONVERTER_OUTPUT_SIZE* property has a different aspect ratio.
- *AMF_VIDEO_CONVERTER_FILL* – Boolean: specifies whether the output image outside the region of interest, which does not fill the entire output surface should be filled with a solid color. The fill color is specified using the *AMF_VIDEO_CONVERTER_FILL_COLOR* property.
- *AMF_VIDEO_CONVERTER_FILL_COLOR* – fill color specified as *AMFColor* to fill the area outside the output rectangle. Applicable only when the *AMF_VIDEO_CONVERTER_FILL* property is set to *true*.
- *AMF_VIDEO_CONVERTER_SCALE* – specifies scaling method. This property can have one of the following values:
    - *AMF_VIDEO_CONVERTER_SCALE_BILINEAR* – use a bilinear scaler
    - *AMF_VIDEO_CONVERTER_SCALE_BICUBIC* – use a bicubic scaler

- *AMF_VIDEO_CONVERTER_FORCE_OUTPUT_SURFACE_SIZE* – instructs the Converter component to use the dimensions of the output surface as output size instead of the size specified by the *AMF_VIDEO_CONVERTER_OUTPUT_SIZE* property when a custom allocator is set through the *AMFComponent::SetOutputDataAllocatorCB* callback.
- *AMF_VIDEO_CONVERTER_COLOR_PROFILE_ENUM* – sets the color profile for color space conversion. This property can be set to one of the following values:
    - *AMF_VIDEO_CONVERTER_COLOR_PROFILE_601* – for ITU-R BT.601 (SDTV), 16..235 color range
    - *AMF_VIDEO_CONVERTER_COLOR_PROFILE_709* – for ITU-R BT.709 (HDTV) , 16..235 color range
    - *AMF_VIDEO_CONVERTER_COLOR_PROFILE_2020* – for ITU-R BT.2020 (UHDTV) , 16..235 color range
    - *AMF_VIDEO_CONVERTER_COLOR_PROFILE_JPEG* – for the full (0..255) color range
    - *AMF_VIDEO_CONVERTER_COLOR_PROFILE_FULL_601* – for ITU-R BT.601 (SDTV), 0..255 full color range
    - *AMF_VIDEO_CONVERTER_COLOR_PROFILE_FULL_709* – for ITU-R BT.709 (HDTV) , 0..255 full color range
    - *AMF_VIDEO_CONVERTER_COLOR_PROFILE_FULL_2020* – for ITU-R BT.2020 (UHDTV) , 0..255 full color range

  The COLOR_PROFILE parameter can fully describe a surface in SDR use case. For HDR use case the TRANSFER_CHARACTERISTIC, COLOR_PRIMARIES and NOMINAL_RANGE parameters describe the surface.
- *AMF_VIDEO_CONVERTER_INPUT_TRANSFER_CHARACTERISTIC* – Characteristic transfer function of the input surface used to perform the mapping between linear light components (tristimulus values) and a nonlinear RGB signal. Used (alongside COLOR_PRIMARIES and NOMINAL_RANGE parameters) to describe surface in HDR use case. See ColorSpace.h for enumeration.
- *AMF_VIDEO_CONVERTER_INPUT_COLOR_PRIMARIES* – Color space primaries for the input surface which are the maximum red, green, and blue value permitted within the color space. Used (alongside TRANSFER_CHARACTERISTIC and NOMINAL_RANGE parameters) to describe surface in HDR use case. See ColorSpace.h for enumeration.
- *AMF_VIDEO_CONVERTER_INPUT_COLOR_RANGE* – Input color range.
  Default = AMF_COLOR_RANGE_UNDEFINED
- *AMF_VIDEO_CONVERTER_INPUT_HDR_METADATA* – AMFBuffer containing AMFHDRMetadata. Default= NULL.
- *AMF_VIDEO_CONVERTER_OUTPUT_TRANSFER_CHARACTERISTIC* – Characteristic transfer function of the input surface used to perform the mapping between linear light components (tristimulus values) and a nonlinear RGB signal. Used (alongside COLOR_PRIMARIES and NOMINAL_RANGE parameters) to describe surface in HDR use case. See ColorSpace.h for enumeration.
- *AMF_VIDEO_CONVERTER_OUTPUT_COLOR_PRIMARIES* – Color space primaries for the input surface which are the maximum red, green, and blue value permitted within the color space. Used (alongside TRANSFER_CHARACTERISTIC and NOMINAL_RANGE parameters) to describe surface in HDR use case. See ColorSpace.h for enumeration.
- *AMF_VIDEO_CONVERTER_OUTPUT_COLOR_RANGE* – Output color range.
  Default = AMF_COLOR_RANGE_UNDEFINED
- *AMF_VIDEO_CONVERTER_OUTPUT_HDR_METADATA* – AMFBuffer containing AMFHDRMetadata. Default= NULL.
- *AMF_VIDEO_CONVERTER_USE_DECODER_HDR_METADATA* – Boolean: Enables use of decoder / surface input color properties above. Default= true

2.3 Submitting Input and Retrieving Output

Once the Converter component is successfully initialized, you may start submitting input samples to it. Input samples must be submitted as *AMFBuffer* objects.

At the same time poll for output by calling *AMFComponent::QueryOutput* on the Converter object. Polling for output samples can be done either from the same thread or from another thread.

Suspend submission of input samples briefly when *AMFComponent::SubmitInput* returns *AMF_INPUT_FULL*. Continue to poll for output samples and process them as they become available.

2.4 Terminating the Converter Component

To terminate the Converter component, call the *Terminate* method, or simply destroy the object. Ensure that the context used to create the Converter component still exists during termination.

3 Sample Applications

A sample application demonstrating the use of the Converter component in AMF is available as part of the AMF SDK in *public/samples/CPPSample/SimpleConverter*. The sample fills 100 frames in a 1920x1080 BGRA surface with an alternating color, submits it as input to the Converter object configured to scale it down to 1280x720 NV12 surface and writes the output to a file.

To run the sample, execute the ‘*SimpleConverter.exe*’ command at the command prompt.