

Advanced Micro Devices

Advanced Media Framework – Video Converter

Programming Guide



Disclaimer

The information contained herein is for informational purposes only, and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information.

Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD's products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale.

AMD, the AMD Arrow logo, ATI Radeon™, CrossFireX™, LiquidVR™, TrueAudio™ and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

Windows™, Visual Studio and DirectX are trademark of Microsoft Corp.



Copyright Notice

© 2014-2016 Advanced Micro Devices, Inc. All rights reserved

Notice Regarding Standards. AMD does not provide a license or sublicense to any Intellectual Property Rights relating to any standards, including but not limited to any audio and/or video codec technologies such as MPEG-2, MPEG-4; AVC/H.264; HEVC/H.265; AAC decode/FFMPEG; AAC encode/FFMPEG; VC-1; and MP3 (collectively, the "Media Technologies"). For clarity, you will pay any royalties due for such third party technologies, which may include the Media Technologies that are owed as a result of AMD providing the Software to you.

MIT license

Copyright (c) 2016 Advanced Micro Devices, Inc. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.



Contents

1	INTRODUCTION	4
2	AMF VIDEO CONVERTER COMPONENT	5
2.1	COMPONENT INITIALIZATION	5
2.2	CONFIGURING THE CONVERTER.....	5
2.3	SUBMITTING INPUT AND RETRIEVING OUTPUT	6
2.4	TERMINATING THE CONVERTER COMPONENT	7
3	SAMPLE APPLICATIONS	8



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.