SAMPLE

Image Overlay using OpenCL

1 Introduction

This sample demonstrates how to build an Image Overlay pipeline using OpenCL kernels. The sample also demonstrates how to use AMD's Unified Video Decoder (UVD) for video decoding and Video Coding Engine (VCE) for video encoding. Color-conversion and image overlaying with and without DX11-OpenCL inter-operations (interops) and DX9-OpenCL inter-operations are also showcased.

2 Using the sample

2.1 Location

`$<installDirectory>\samples\mediaFoundation\imageOverlayOpenCl\`

2.2 Contents

Package contents

Folder: `$<installDirectory>\samples\mediaFoundation\imageOverlayOpenCl\src\`

File name	Description
`ImageOverlayConfig.cpp`	Contains function to parse the user specified configuration file
`ImageOverlayMain.cpp`	Demonstrates how to invoke & execute the Image Overlay pipeline
`MftAsyncColorConvert.cpp`	Implementation of AsyncTransform class for Color Convert component
`MftAsyncTransform.cpp`	Implementation of AsyncTransform class for Overlay component
`MftDx9ColorConvert.cpp`	Functions implementing Color Conversion with & without Dx9-OpenCl Interop
`MftDx9DllColorConvertMain.cpp`	Implements creation of Color Conversion MFT instance on DX9
`MftDx9DllMain.cpp`	Implements creation of Image Overlay MFT instance on DX9
`MftDx9Overlay.cpp`	Functions implementing Image Overlay with and without Dx9-OpenCl Interop
`MftDx11ColorConvert.cpp`	Functions implementing Color Conversion with and without Dx11-OpenCl Interop
`MftDx11DllColorConvertMain.cpp`	Implements creation of Color Conversion MFT instance on DX11
`MftDx11DllMain.cpp`	Implements creation of Image Overlay MFT instance on DX11
`MftDx11Overlay.cpp`	Functions implementing Image Overlay with and without Dx11-OpenCl Interop
`TranscodeSession.cpp`	Contains functions to Create and Instantiate MFTs and to Build, Load and Execute the Topology

Folder:
$<*installDirectory*>\samples\mediaFoundation\imageOverlayOpenCl\inc\

File name	Description
cl_d3d11.h	OpenCL D3D11 header file
cl_dx9_media_sharing.h	OpenCL D3D9 header file
ImageOverlayApi.h	Contains declaration for Image Overlay MFT pipeline preparation and run functions
ImageOverlayConfig.h	Contains declaration for configuration file reading functions
MftAsyncColorConvert.h	Async Color Convert MFT class declaration
MftAsyncTransform.h	Async Image Overlay MFT class declaration
MftDx9ColorConvert.def	A module-definition file containing module statements that describe attributes of Color Convert DLL on Dx9
MftDx9ColorConvert.h	Defines class for Color Converting on DX9
MftDx9Copy.cl.h	OpenCl kernel to copy RGBA image in DX9 domain
MftDx9NV12ToRGBA.cl.h	OpenCl kernel to convert NV12 Image to RGBA Image in DX9 domain
MftDx9Overlay.cl.h	OpenCl kernel to Overlay RGBA image in DX9 domain
MftDx9Overlay.h	Defines class for Image Overlaying on DX9
MftDx9RGBAToNV12.cl.h	OpenCl kernel to convert RGBA Image to NV12 Image in DX9 domain
MftDx9Transform.def	A module-definition file containing module statements that describe attributes of Image Overlay DLL on Dx9
MftDx9Transform.h	Async transform create instance function declaration on DX9
MftDx11ColorConvert.def	A module-definition file containing module statements that describe attributes of Color Convert DLL on Dx11
MftDx11ColorConvert.h	Defines class for Color Converting on DX11
MftDx11Copy.cl.h	OpenCl kernel to copy RGBA image in DX11 domain
MftDx11NV12ToRGBA.cl.h	OpenCl kernel to convert NV12 Image to RGBA Image in DX11 domain
MftDx11Overlay.cl.h	OpenCl kernel to Overlay RGBA image in DX11 domain
MftDx11Overlay.h	Defines class for Image Overlaying on DX11
MftDx11RGBAToNV12.cl.h	OpenCl kernel to convert RGBA Image to NV12 Image in DX11 domain
MftDx11Transform.def	A module-definition file containing module statements that describe attributes of Image Overlay DLL on Dx11
MftDx11Transform.h	Async transform create instance function declaration on DX11
OpenCL.bmp	Overlay Image
Resource.rc	Resource-definition script that describes the resources (bitmap) used by the sample
TranscodeSession.h	Defines class for transcode session
VideoEffect.h	Defines function structure for common video effect API functions

Folder:
$<*installDirectory*>\samples\mediaFoundation\imageOverlayOpenCl\config\

File name	Description
exampleConfig.cfg	Configuration file containing the H.264 encoder and common configuration parameters

Folder:
$<*installDirectory*>\samples\mediaFoundation\imageOverlayOpenCl\docs\

File name	Description
MediaSDK_MFT_imageOverlayOpenCL.pdf	Sample documentation

Folder:
$<*installDirectory*>\samples\mediaFoundation\simpleTranscode\build\windows\

File name	Description
imageOverlayOpenClVs10.sln	Visual Studio 10 Image Overlay solution file
imageOverlayOpenClVs10.vcxproj	Visual Studio 10 Image Overlay project file
imageOverlayOpenClVs10.vcxproj.filters	Visual Studio 10 Image Overlay project filter file
imageOverlayOpenClVs12.sln	Visual Studio 12 Image Overlay solution file
imageOverlayOpenClVs12.vcxproj	Visual Studio 12 Image Overlay project file
imageOverlayOpenClVs12.vcxproj.filters	Visual Studio 12 Image Overlay project filter file
mftDx9ColorConvertOpenClVs10.vcxproj	Visual Studio 10 project file for building custom D3D9 color convert MFT
mftDx9ColorConvertOpenClVs10.vcxproj.filters	Visual Studio 10 D3D9 color convert project filter file
mftDx9ColorConvertOpenClVs12.vcxproj	Visual Studio 12 project file for building custom D3D9 color convert MFT
mftDx9ColorConvertOpenClVs12.vcxproj.filters	Visual Studio 12 D3D9 color convert project filter file
mftDx9OverlayOpenClVs10.vcxproj	Visual Studio 10 project file for building custom D3D9 Image Overlay MFT
mftDx9OverlayOpenClVs10.vcxproj.filters	Visual Studio 10 D3D9 Image Overlay project filter file
mftDx9OverlayOpenClVs12.vcxproj	Visual Studio 12 project file for building custom D3D9 Image Overlay MFT
mftDx9OverlayOpenClVs12.vcxproj.filters	Visual Studio 12 D3D9 Image Overlay project filter file
mftDx11ColorConvertOpenClVs10.vcxproj	Visual Studio 10 project file for building custom D3D11 color convert MFT
mftDx11ColorConvertOpenClVs10.vcxproj.filters	Visual Studio 10 D3D11 color convert project filter file
mftDx11ColorConvertOpenClVs12.vcxproj	Visual Studio 12 project file for building custom D3D11 color convert MFT
mftDx11ColorConvertOpenClVs12.vcxproj.filters	Visual Studio 12 D3D11 color convert project filter file
mftDx11OverlayOpenClVs10.vcxproj	Visual Studio 10 project file for building custom D3D11 Image Overlay MFT
mftDx11OverlayOpenClVs10.vcxproj.filters	Visual Studio 10 D3D11 Image Overlay project filter file
mftDx11OverlayOpenClVs12.vcxproj	Visual Studio 12 project file for building custom D3D11 Image Overlay MFT

mftDx11OverlayOpenClVs12.vcxproj.filters	Visual Studio 12 D3D11 Image Overlay project filter file

2.3 Parameters

Encoder-specific configuration parameters

Parameter name	Default value	Supported range	Remarks
encGOPSize	20		Max number of frames in a GOP (0=auto)
encMeanBitrate	3000000		Bitrate of encoded video (bits per second)
encMaxBitrate	4000000		Maximum bitrate of encoded video (used only for VBR) in bits per second
encBufferSize	2000000		VBR buffer size
encNumBFrames	1	0 - 3	Specifies the number of B frames to be inserted
encCompressionStandard	77	For supported values, see http://msdn.microsoft.com/en-us/library/windows/desktop/dd318776(v=vs.85).aspx	Compression standard
encRateControlMethod	1	eAVEncCommonRateControlMode_CBR = 0, eAVEncCommonRateControlMode_PeakConstrainedVBR = 1, eAVEncCommonRateControlMode_UnconstrainedVBR = 2, eAVEncCommonRateControlMode_Quality = 3	For more details, see http://msdn.microsoft.com/en-us/library/windows/desktop/dd388772(v=vs.85).aspx
encLowLatencyMode	0	1 - True 0 - False	Specifies whether the output stream should be structured so that the encoded stream has a low decoding latency.
encQualityVsSpeed	60	0 - Low quality faster encoding 100 - Higher quality, slower encoding	
encCommonQuality	50	0 to 100 0 - low quality 100 - highest quality	This parameter is used only when encRateControlMethod is set to eAVEncCommonRateControlMode_Quality. in this mode the encoder selects the bit rate to match the quality settings.

Common configuration parameters

Parameter name	Default value	Supported range	Remarks
`useInterop`	1	Enable=1 Disable=0	Enables/Disables the use of OpenCL Interops.
`useSWCodec`	0	Enable=1 Disable=0	Enables/Disables the use of software Encoder and Decoder. If set to 0, hardware-based codecs are used to encode and decode the stream; otherwise, software-based codecs are used.

2.4 Compile

1. Ensure that the following tools and SDKs are present:
   - Microsoft Visual Studio 2010 or 2012
     If Windows Software Development Kit (SDK) is not installed, install it from http://msdn.microsoft.com/en-us/library/windows/desktop/hh852363.aspx.
2. Open one of the following solution files:
   `$<installDirectory>\samples\mediaFoundation\imageOverlayOpenCl\build\windows\imageOverlayOpenClVs12.sln`
   `$<installDirectory>\samples\mediaFoundation\imageOverlayOpenCl\build\windows\imageOverlayOpenClVs10.sln`
3. Build the sample:
   - Open the `imageOverlayOpenClVs10.sln` solution file with Microsoft Visual Studio 2010 Professional Edition or the `imageOverlayOpenClVs12.sln` solution file with Microsoft Visual Studio 2012 Professional Edition.
   - To build all the solutions, select `Build > Build Solution`.
   - Select the project file in the Solutions Explorer.
     The executable `imageOverlayOpenCl.exe` is created in the following folders for 32-bit builds and 64-bit builds:
     `$<installDirectory>\samples\mediaFoundation\imageOverlayOpenCl\bin\x86\`
     `$<installDirectory>\samples\mediaFoundation\imageOverlayOpenCl\bin\x86_64\`

3 Run

The sample can be executed using either software-based or hardware-based codecs. If hardware codecs are to be used and if the sample is executed on an AMD platform then the AMD VCE and UVD hardware blocks are used; otherwise, software-based codecs are used to decode and encode the stream.

On the command prompt, change to the directory that contains the executable, and execute the following command:
`imageOverlayOpenCl.exe -i <input.avi> -o <output.asf> -c <configfile> -l <0, 1, or 2>`

-l enables the logging. Setting "0" means no logging. "1" generates the log at the API level. "2" generates logs at the video edit session level.

The `$<installDirectory>\inc\ErrorCodes.h` file contains information about the error codes. You can also print the Microsoft error codes by using the `getMicrosoftErrorCode()` API, as shown in `ImageOverlayMain.cpp`. The Microsoft error codes can be understood from the `Mferror.h` file that Microsoft provides as part of its Windows kits.

The configuration file associated with this sample is located in the following folder:
`$<installDirectory>\samples\mediaFoundation\imageOverlayOpenCl\config\`

The `useSWCodec` flag is used to select either software- or hardware-based codecs and the `useInterop` flag is used to either enable or disable the use of DX-OpenCL interops.

The output file containing the overlayed image is generated in the path specified by the user.

4 Implementation Details

The sample implements the following image overlay pipeline.

In addition to the Video Decoder and Video Encoder MFTs, the topology includes two transform modules:

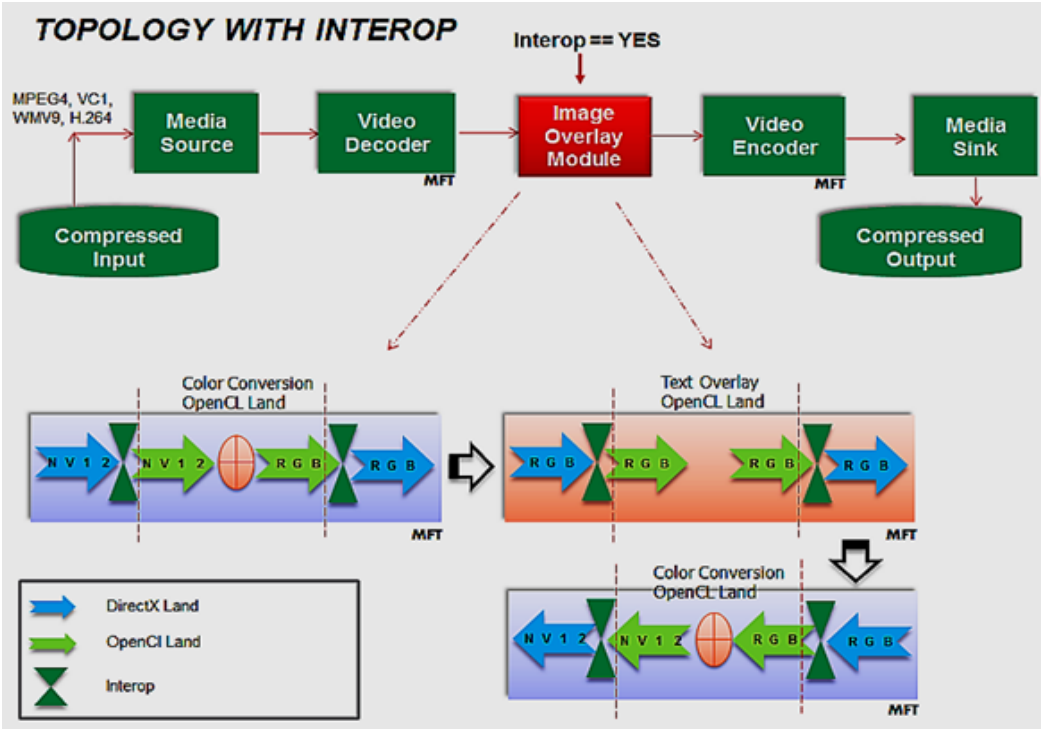
- Color Conversion
- Image Overlay Video Filtering

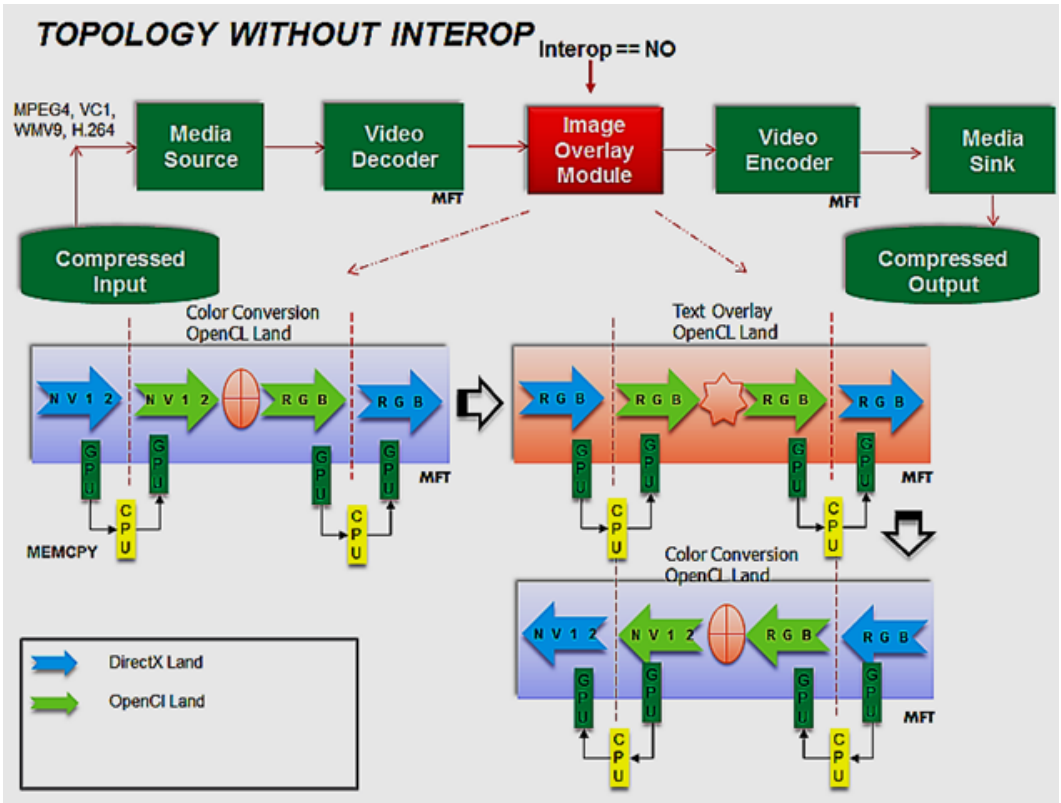
The above MFTs perform the desired functionality in OpenCL land. A run-time switch is used to enable/disable the Interop, which:

- if enabled, involves translating the input from DirectX to OpenCL and the generated output from OpenCl to DirectX land. This results in performance improvement as there are no memory copy of buffers from the GPU to the CPU and back to the GPU.
- if disabled, involves a copy of the buffer from the GPU to the CPU and back to the GPU for access of buffers from DirectX to OpenCL and from OpenCL to DirectX land.

The output from the decoder is NV12 content which is color-converted to RGBA using the Color conversion MFT. RGBA data is then fed to the Overlay MFT to overlay and then color-converted to NV12 before feeding the overlayed imaged to the Encoder.

The following two diagrams explain the data flow for the two scenarios, i.e., with and without inter-operations.





On Windows 8/8.1, the `DXGIDeviceManager` class and its associated object, `D3D11Texture2D`, are used. The video decoder MFT provides the `D3D11Texture2D` object with MFVideoFormat_NV12-decoded video as the output. The OpenCL Image Overlay MFT uses the DirectX 11-OpenCL interop to obtain the OpenCL image from `D3D11Texture2D`, on which the image overlay kernel is applied.

On Windows 7, the `Direct3DDeviceManager` and its associated object, `Direct3DSurface9`, are used. The video Decoder MFT provides the `Direct3DSurface9` object with MFVideoFormat_NV12-decoded video as the output. The OpenCL Image Overlay MFT uses the DirectX9-OpenCL interop to obtain the OpenCL image from `Direct3DSurface9`, on which the image overlay kernel is applied.

By default, the sample uses the DirectX 11-OpenCL interop path on Windows 8/8.1 and the DirectX9-OpenCL interop path on Windows 7. The DirectX 11-OpenCL interop is not supported on Windows 7.

5 Supported formats

The following file formats are supported:

- Input file/container formats: .avi, .mp4, .wmv
- Video decoders supported: H264, MPEG4 Part II, WMV9, VC1
- Output file/container format: .asf
- Video encoder supported: H264

6 Known limitations

The sample is currently supported on the following platforms:

- Windows 7 (DirectX 9)
- Windows 8/8.1 (DirectX 9 and DirectX 11)

Contact

Advanced Micro Devices, Inc.
One AMD Place
P.O. Box 3453
Sunnyvale, CA, 94088-3453
Phone: +1.408.749.4000

For AMD Accelerated Parallel Processing:

URL: developer.amd.com/appsdk
Developing: developer.amd.com/
Forum: developer.amd.com/openclforum