SAMPLE

Pipeline Playback

1 Introduction

The purpose of this sample is to demonstrate how to build and execute a pipeline-based video playback using AMD Media Framework (AMF). The sample decodes H.264 elementary stream and displays the generate YUV frames.

2 Using the sample

2.1 Location

`$<installDirectory>\samples\amf\pipelinePlayback\`

2.2 Contents

Package Contents

Folder:
`$<installDirectory>\samples\amf\pipelinePlayback\src\`

File name	Description
`PlaybackPipeline.cpp`	Source file for Playback Pipeline class application
`PlaybackMain.cpp`	Source file for Playback window application

Folder:
`$<installDirectory>\samples\amf\pipelinePlayback\inc\`

File name	Description
`PlaybackPipeline.h`	Header file for Playback Pipeline class
`PlaybackMain.h`	Header file for playback window application

Folder:
`$<installDirectory>\samples\amf\pipelinePlayback\build\windows\`

File name	Description
`PipelinePlaybackVs10.sln`	Microsoft Visual Studio 10 solution file
`PipelinePlaybackVs10.vcxproj`	Microsoft Visual Studio 10 project file
`PipelinePlaybackVs10.vcxproj.filter`	Microsoft Visual Studio 10 project filter file
`PipelinePlaybackVs12.sln`	Microsoft Visual Studio 12 project solution file
`PipelinePlaybackVs12.vcxproj`	Microsoft Visual Studio 12 project file
`PipelinePlaybackVs12.vcxproj.filter`	Microsoft Visual Studio 12 project filter file

Folder:
$<installDirectory>\samples\amf\common\src\

File name	Description
BitStreamParser.cpp	Source file for parsing bit-stream
BitStreamParserH264.cpp	Source file for H264 Bit-stream Parser
CmdLogger.cpp	Source file for Command Logging
DeviceDX9.cpp	Source file for DX9 Device
DeviceDX11.cpp	Source file for DX11 Device
ParametersStorage.cpp	Source file for Parameters Storage
Pipeline.cpp	Source file for the Pipeline
PlatformWindows.cpp	Source file for Platform Windows
Thread.cpp	Source file for Thread creation and handling
VideoPresenter.cpp	Source file for Video Presenter
VideoPresenterDX9.cpp	Source file for DX9 Video Presenter
VideoPresenterDX11.cpp	Source file for DX11 Video Presenter

Folder:
$<installDirectory>\samples\amf\common\inc\

File name	Description
AMFPlatform.h	Header file for Platform
BitStreamParser.h	Header file for parsing bit-stream
BitStreamParserH264.h	Header file for H264 Bit-stream Parser
ByteArray.h	Header file for Byte Array Processing
CmdLogger.h	Header file for Command Logging
DeviceDX9.h	Header file for DX9 Device
DeviceDX11.h	Header file for DX11 Device
ParametersStorage.h	Header file for Parameters Storage
Pipeline.h	Header file for the Pipeline
PipelineElement.h	Header file for Pipeline Element
PlatformWindows.h	Header file for Platform Windows
Thread.h	Header file for Thread creation and handling
VideoPresenter.h	Header file for Video Presenter
VideoPresenterDX9.h	Header file for DX9 Video Presenter
VideoPresenterDX11.h	Header file for DX11 Video Presenter

Folder:
$<*installDirectory*>\samples\amf\pipelinePlayback\res\

File name	Description
resource.h	Resource header file
PipelinePlayback.ico	Icon file for the Playback application
PipelinePlayback.rc	Resources used by the Playback application

small.ico	Small icon file for the Playback application
stdafx.cpp	Source file for the STD AFX file
stdafx.h	Header file for the STD AFX file
targetver.h	Target version header file

Folder:
$<*installDirectory*>\samples\amf\pipelinePlayback\docs\

File name	Description
MediaSDK_AMF_pipelinePlayback.pdf	Sample documentation

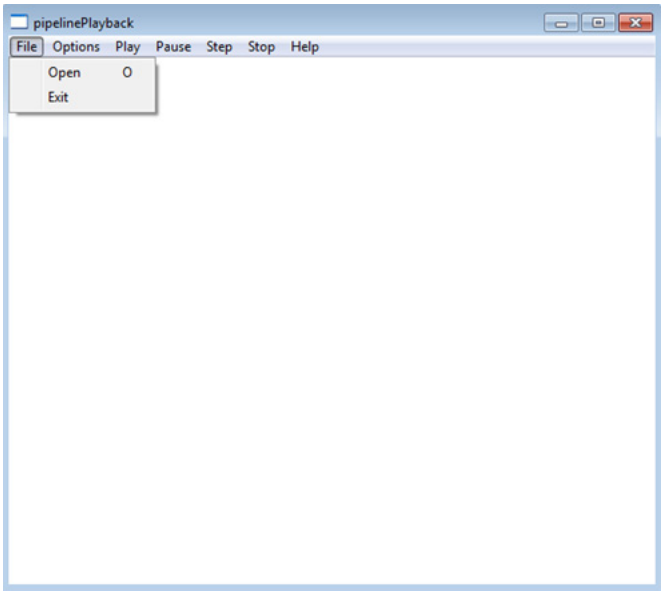
2.3 Compile

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:
   - `$<installDirectory>\samples\amf\pipelinePlayback\build\windows\pipelinePlaybackVs12.sln`
   - `$<installDirectory>\samples\amf\pipelinePlayback\build\windows\pipelinePlaybackVs10.sln`
3. Build the sample:
   - Open the `pipelinePlaybackVs10.sln` solution file with Microsoft Visual Studio 2010 Professional Edition or the `pipelinePlaybackVs12.sln` solution file with Microsoft Visual Studio 2012 Professional Edition.
   - To build all the solutions, select `Build > Build Solution`.
   - The executable `pipelinePlayback.exe` is created in the following folders for 32-bit builds and 64-bit builds:
     `$<installDirectory>\samples\amf\pipelinePlayback\bin\x86\`
     `$<installDirectory>\samples\amf\pipelinePlayback\bin\x86_64\`
   - Depending on the build (i.e. 32-bit or 64-bit), the custom build step copies the appropriate `.dlls` file from the `$<installDirectory>\dll\amf\` folder into the relevant `\bin\` directory.

3 How to Run

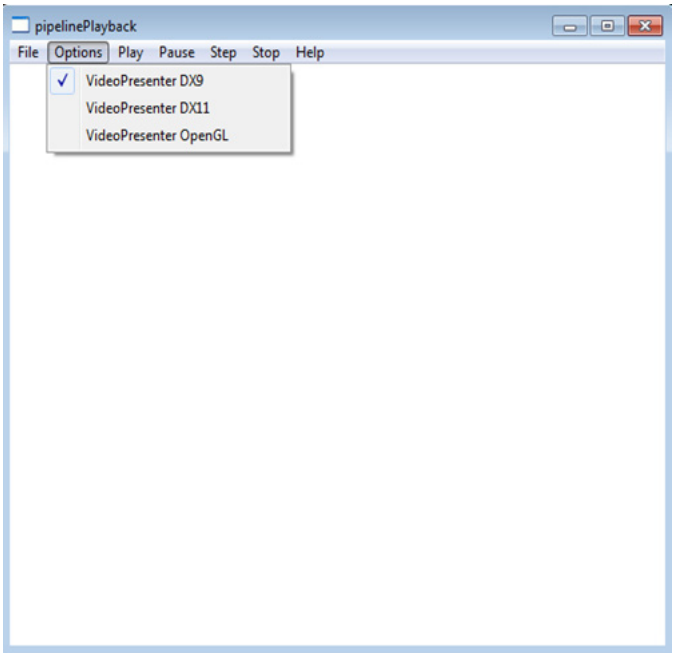
The sample can be executed on an AMD platform that includes the VCE and UVD hardware blocks.

Double-click the generated executable; alternatively, in Microsoft Visual Studio, press CTRL + F5. The following screen is displayed.



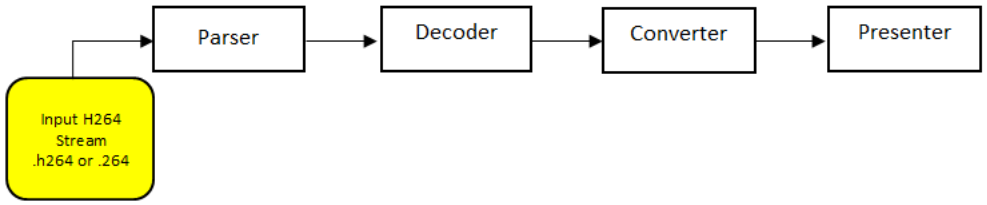
`File > Open`: Use this option to browse an input H.264 Elementary stream
`Play`: Use this option to Play the Input File
`Pause`: Use this option to Pause the playback
`Stop`: Use this option to Stop the Playback
`Options`: This option is used to select the following:

Memory Type: Selects the Memory Type from `Video Presenter DX9` or `Video Presenter DX11` or `Video Presenter OpenGL`



4 Implementation Details

The sample implements the following playback pipeline:



The Data in the Decode Pipeline flows through the following processing elements:

- Parser: H.264 Elementary stream data is first read by the parser which finds the SPS & PPS, finds NALU unit, and populate data structures which are fed to the H.264 Decoder.
- Decoder: HW Accelerated (UVD) H.264 Video Decoder. Decodes the input content to generate NV12 frames.
- Converter: Component which
  - Converts the decoded NV12 output format and memory type to the display BRGA color format and user-specified presenter memory type. Presenter memory type is specified by the user in the Playback window under the `Options` tab.
  - Resizes the decoded output width and height to the dimensions of the display window
- Presenter: Presents the decoded and converted frames to the display window.

5 Supported formats

The following file formats are supported:

- Input file/container formats: Elementary H264 Stream
- Video decoders supported: H.264

6 Known Limitations

The sample is currently supported on the following platforms:

- Windows 7 (DirectX 9)
- Windows 8.1 (DirectX 9 and DirectX 11.1)

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