Media Samples Guide

Overview

Samples work with **Intel® Media Server Studio 2017 - SDK for Linux* Server** (hereinafter referred to as "**SDK**").

They demonstrate how to incorporate the **SDK** API into various applications.

Some samples can work with **Intel® Media Server Studio – HEVC Decoder & Encoder** from Intel® Media Server Studio - Professional Edition (hereinafter referred to as "**HEVC Encoder**", "**HEVC Decoder**", "**HEVC**").

Not all of the samples listed below might be applicable and supported for a particular product. Make sure to check the respective release notes document for potential limitations.

What's New

- Conditional compilation markup is added. Now samples can be compiled with dispatcher library made for previous API versions. This samples package supports API 1.23, 1.22 and 1.21 (some features will be disabled if previous API is used).
- sample_encode is extended with new options:
  - -signal option is added, it sets transfer matrix coefficients for mfxExtVideoSignalInfo structure.
  - -dump option is added. It allows to print full configuration of encoder object into file (all configuration structures are printed in human-readable form).
- sample_multi_transcode is extended with new options:
  - -i::rgb4_frame - new input type for creating alpha-blended overlays. This input file should contain one RGB4 frame with alpha channel.
  - -f - for setting new framerate for FRC and deinterlace.
  - -override_decoder_framerate and -override_encoder_framerate - for explicit overriding decoder and encoder framerates.
  - -vpp_comp_dump - saves non-compressed data directly to the file or just drops the data if null_render is set instead of file name
- Explicit composition tiling support is added to the sample_multi_transcode (-vpp_comp_tile_id and -vpp_comp_num_tiles).
- FEI Encoding Sample is added to samples package (it was distributed separately before).

Package contents

Full list of available samples:

- **Video Decoding Sample**

  Console application which performs decoding of elementary compressed video stream to raw frames. Includes the following features:

  - stereoscopic 3D (S3D) rendering of elementary MVC (Multi-View Video Coding) streams
  - decoding of HEVC (High Efficiency Video Coding) video via **HEVC Decoder**

---

* Other names and brands may be claimed as the property of others.

  - decoding with video post processing (color conversion) of raw video sequences
  - screen capturing via screen capture plugin
- **Video Encoding Sample**

  Console application which performs encoding of raw video frames into elementary compressed stream. Includes the following features:

  - video resizing
  - video rotation via User Plug-in Sample
  - video rotation via User Plug-in Sample using OpenCL™
  - encoding HEVC video via **HEVC Encoder**
- **Video Processing Sample**

  Console application which performs various video processing algorithms on raw frames.
- **Video Transcoding Sample**

  Console application which performs transcoding of elementary video stream from one compressed format to another. Includes the following features:

  - multiple video streams transcoding
  - video resizing, de-interlacing
  - video rotation via User Plug-in Sample
  - video rotation via User Plug-in Sample using OpenCL
  - video processing using VPP algorithms
- **OpenCL Video Motion Estimation Sample and OpenCL Advanced Video Motion Estimation Samples**

  Console application which provides step-by-step guidelines on the using Intel's motion estimation extension for OpenCL standard. The motion estimation extension includes a set of host-callable functions for frame-based Video Motion Estimation.
- **FEI Encoding Sample**

  Console application that uses SDK FEI API (Flexible Encoder Interface) and demonstrates capability to stream internal encoder information during encoding process to specified output. Intel® Media Server Studio 2017R3 - SDK for Linux* Server is required for this sample.

  Supported interfaces are:

  - PreENC
  - FEI ENCODE
  - ENC
  - PAK
  - DecodeStreamOut

Each sample includes:

- a readme file for each sub-sample
- source and header files for each sub-sample

Samples package has one installer for all sub-samples.

Software & Hardware Requirements

Hardware:

- Hardware requirements are the same as described in **SDK** Release Notes
- (Optional) HDMI* 1.4, eDP* 1.1 or similar based monitor/TV as primary display
- (Optional) Active shutter glasses

Software:

- See <msdk_install-folder>/media_server_studio_sdk_release_notes.pdf for **SDK** general requirements. To build **Samples** you additionally need the following components to be installed and properly configured on the system:
- **For CentOS*:**

```
$ sudo yum install gcc g++ make cmake perl libX11-devel mesa-libGL-devel
```

- Samples can be built with GCC/G++ compiler version 4.6 and CMake* version 2.8.0 or higher.
- For samples with OpenCL (**Video Encoding, Video Transcoding, Video Motion Estimation, Interoperability**) it is required to install **Intel® Media Server Studio – Intel® SDK for OpenCL™ Applications** and **Intel® Media Server Studio – Graphics Drivers**.

Build Instructions

To build samples the following environment variable should be setup:

```
$ export MFX_HOME=/mediasdk/installation/folder
```

Go to the samples directory and execute build.pl script without arguments to see the help:

```
$ ./build.pl
Copyright (c) 2014 Intel® Corporation. All rights reserved.
This script performs Samples projects creation and build.
Usage: perl build.pl --cmake=ARCH,GENERATOR,CONFIG [--clean] [--build]
Possible variants:
 ARCH = intel64
 GENERATOR = make
 CONFIG = debug | release
Environment variables:
 MFX_HOME=/path/to/mediasdk/package # required
 MFX_VERSION="0.0.000.0000" # optional
Optional flags:
 --clean - clean build directory before projects generation / build
 --build - try to build projects before generation (requires
cmake>=2.8.0)
Examples:
 perl build.pl --cmake=intel64,make,debug [ only
generate projects ]
 perl build.pl --cmake=intel64,make,debug --build [ generate
and then build ]
 perl build.pl --cmake=intel64,make,debug --build --clean [ generate,
clean and build ]
```

Note that optional flag --enable-x11 temporary does not affect compilation process (temporary limitation) - so you should have libx11-devel installed, as it is described in Software and Hardware Requirements section.

Script invokes specified CMake* projects generator and optionally builds them (option available for cmake>=2.8.0). At the moment only make files generator for UNIX-like systems is supported. Project files will be placed in the folder named by the requested configuration; for example:

```
/__cmake
intel64.make.release
intel64.make.debug
```

To build generated project files use generator-specific approaches. For example, to build samples from make files invoke:

```
$ make -C <install-folder>/__cmake/intel64.make.release
```

With CMake older than 2.8.0 all samples can be built at once with the following command:

```
$ ./build.pl --cmake=intel64,make,release --clean --build
```

Binaries will appear in the following folder:

```
$ ls -1 __cmake/intel64.make.release/__bin/release/
sample_decode
sample_encode
sample_multi_transcode
sample_vpp
```

Running the Software

DRM backend specific notes:

- For application to work thru DRM application should be authorized to access graphics card. VA-API DRM backend supports 2 authentication models:
  - The first model can be applied on the system with no installation of Graphic Server. In this case you need root privileges to run:

    ```
    $ sudo LD_LIBRARY_PATH=$MEDIASDK_INSTALL_FOLDER/bin/x64 \
    $ sample_decode h264 -i input.264 -o output.yuv –vaapi –hw
    ```

  - The second model assumes that X server is installed and running. In this case DRM authentication will actually go thru LibVA X11 backend and, thus, thru X server which already has access to the graphic card. The only thing user should be sure in is that he is logged on to the X server (or has access) and `DISPLAY` environment variable is set properly. For example:

    ```
    $ export DISPLAY=:0.0
    $ sudo LD_LIBRARY_PATH=$MEDIASDK_INSTALL_FOLDER/bin/x64 \
    $ sample_decode h264 -i input.264 -o output.yuv –vaapi –hw
    ```

- It can be noted that DRM-itself authentication can still be tried out even with running X server, but you need to remove `DISPLAY` environment variable and use root privileges:

  ```
  $ export -n DISPLAY
  $ sudo LD_LIBRARY_PATH=$MEDIASDK_INSTALL_FOLDER/bin/x64 \
  $ sample_decode h264 -i input.264 -o output.yuv –vaapi –hw
  ```

X11 backend specific notes:

- To use this backend user should be sure that he is logged into X server or is allowed to make connections to the X server
- If user is allowed to use X and logged into machine remotely (thru SSH) he needs `DISPLAY` environment variable properly set. For example:

  ```
  $ export DISPLAY=:0.0
  $ sample_decode h264 -i input.264 -o output.yuv –vaapi –hw
  ```

License

This software is distributed under the BSD-3 clause license, full text of license is reproduced below:

Legal Information

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or by visiting *Intel's Web Site*.

MPEG is an international standard for video compression/decompression promoted by ISO. Implementations of MPEG CODECs, or MPEG enabled platforms may require licenses from various entities, including Intel Corporation.

Intel, the Intel logo, Intel Core are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel.

Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804
