Decoding Sample

Overview

Decoding Sample works with **Intel® Media Server Studio 2017 for Linux** (hereinafter referred to as "**SDK**").

It demonstrates how to use the **SDK** API to create a sample console application that performs decoding of various video compression formats.

The sample can work together with **Intel® Media Server Studio – HEVC Decoder & Encoder** (hereinafter referred to as "**HEVC**").

Note: To run HEVC, please read the instructions in the “HEVC Plugin” section carefully.

Features

Decoding Sample supports the following video formats:

Format type	
input (compressed)	H.264 (AVC, MVC – Multi-View Coding), MPEG-2 video, VC-1, JPEG*/Motion JPEG, HEVC (High Efficiency Video Coding), VP8
output (uncompressed)	YUV420

Note 2: Decoding Sample renders the decoded video stream to a file in YUV 4:2:0 sampling format, with the color planes Y, U and V in that order.

Hardware Requirements

See `<install-folder>\Media Samples Guide.pdf`.

Software Requirements

See `<install-folder>\Media Samples Guide.pdf`.

How to Build the Application

See `<install-folder>\Media Samples Guide.pdf`.

Running the Software

See `<install-folder>\Media Samples Guide.pdf`.

The executable file requires the following command-line switches to function properly:

h265|h264|mpeg2|vc1| vp8|mvc|jpeg|capture	Input video type. This is an elementary video stream. The use of option h265 or capture is possible only if corresponding plugins are installed.
-i <InputFile>	Input (compressed) video file, name and path.
-o <Output>	Specifies output (YUV) video file(s), name and path. With MVC, specify the file name without extension to use as a pattern. The sample creates several output files with names such as “filename_N.yuv” according to the number of views in the MVC stream.

The following command-line switches are optional:

-p guid	32-character hexadecimal guid string. Optional for in-box plugins, required for user-decoder ones (HEVC, f.e.).
-path path_to_plugin	Path to decoder plugin (works only in pair with ‘`-p`’ option and requires guid to be specified).
-vaapi	Use VAAPI surfaces
-r	Enables on-screen rendering support under X Server (requires X Server running)
-rdrm	Enables on-screen rendering support thru the framebuffer from the Linux Console mode (requires any graphical server, like X server, to be switched off)
-rdrm-<connector>	Same as -rdrm, but enables on-screen rendering support to the monitor connected thru the specified <conector>. Some possible connectors are: DisplayPort, HDMIA, HDMIB, VGA, DVII, DVID, DVIA, eDP and others
-perf	Turn on asynchronous flipping for Wayland rendering.
-window <x> <y> <w> <h>	Set render window position and size.
-hw	Use platform-specific implementation of **SDK** (default)
-sw	Use software implementation of **SDK** (platform-specific implementation is used by default)
-di <bob or adi>	Enable deinterlacing:BOB or ADI
-w	Output width
-h	Output height
-jpeg_rgb	Set JPEG Color format to RGB4
-nv12, -i420, -rgb4, -p010, -a2rgb10	Output color format (native decoder's output format by default). Note that in case of -i420 option, pipeline uses nv12 color format for output (surfaces format), but during file writing data is converted into i420.
-rgb4_fcr	Set pipeline output format and output file format to RGB4 in full color range
-low_latency	Configures **Decoding Sample** for low latency mode
-calc_latency	Calculate per frame decoding latency
-w	Output width

-h	Output height
-threads_num	Number of mediasdk task threads
-threads_schedtype	Scheduling type of mediasdk task threads
-threads_priority	Priority of mediasdk task threads
-gpucopy::<on,off>	Enable/disable GPU Copy functionality
-timeout	Decode in a loop not less than specific time in seconds. Performs complete input stream decoding on every iteration. Output file frames amount can be bigger than in input due to buffered frames in decoder. Output file is rewrote every iteration.
-?	Print help

Note 1: You need to have **HEVC** installed to run with h265 codec. In case of HW library (-sw key is not specified) it will firstly try to load HW plugin, in case of failure - it will try SW one if available.

Below are examples of a command-line to execute **Decoding Sample**:

```
$ sample_decode h264 -i input.h264 -o output.yuv -hw
```

Please, also pay attention on "Running the Software" section of `<install-folder>/Media Samples Guide.pdf` document where you will find important notes on backend specific usage (drm and x11).

HEVC Plugin

HEVC codec is implemented as a plugin unlike codecs such as MPEG2 and AVC. We provide multiple implementations of the HEVC plugin for Decode and Encode – SW, HW and GPU-accelerated. In our samples, depending on the underlying platform, the HW plugin is loaded. If the HW plugin is not supported, the SW plugin gets loaded.

Note 1: The HEVC SW and GPU-accelerated plugins (HEVC Decode SW, HEVC Encode SW, and HEVC Encode GPU-accelerated) are available in the HEVC package which is part of the Intel® Media Server Studio Professional Edition. You can find the available plugins and their IDs from `$MFX_ROOT/include/mfxplugin.h` file.

Note 2: HW Accelerated HW plugins for HEVC Encode and Decode are supported from 6th Generation Intel CoreTM Processors with Intel IrisTM Pro Graphics or Intel HD Graphics 6000/7000+ Series (codename Skylake).

For previous generations (4th and 5th Generation Intel CoreTM Processors with Intel IrisTM Pro Graphics or Intel HD Graphics 4200+ Series), the HEVC SW and GPU-accelerated versions are supported.

Note 3: HEVC Encode has the GPU-accelerated (henceforth referred to as GACC) plugin. To load the plugin, you have to explicitly specify the plugin ID "e5400a06c74d41f5b12d430bbaa23d0b" using the "-p" parameter.

Our samples load the SW HEVC plugins, unless the GUI for the GACC counterparts are specified using "-p" parameter. For example, the following command-lines will use the SW HEVC Decode and Encode plugin respectively:

```
$ sample_decode h265 -i input.265 -o output.yuv
```

```
$ sample_encode h265 -i input.yuv -o output.h265 –w 720 –h 480 –b 10000 –f
 30 –u quality
```

(Note: HEVC Decode GACC plugin is not available on Linux).

You can enforce a plugin to be loaded by specifying the plugin ID for the same. For example, the below command for sample_decode will load the HW HEVC plugin, and the sample_encode will load the GPU-accelerated plugin:

```
$ sample_decode h265 -i input.265 -o output.yuv -p
 33a61cb4c27454ca8d85dde757c6f8e
```

```
$ sample_encode h265 -i input.yuv -o output.h265 -w 720 -h 480 -b 10000 -p
 e5400a06c74d41f5b12d430bbaa23d0b
```

Known Limitations

- **Decoding Sample** does not fully decode some video streams from a networked folder. Instead, copy the input file to local storage prior to decoding.
- **Decoding Sample** renders output in the simplest way. The rendering window does not support time stamps, aspect ratio.
- Decoding Sample cannot render P010 streams in case of SW library usage.
- `-low_latency` and `-calc_latency` options should be used with H.264 streams having exactly 1 slice per frame. Preferable streams for an adequate latency estimate are generated by **Conferencing Sample**. The options are also effective for JPEG* input streams. For all other input formats application would return an error.
- If overlay is not supported by your hardware or software you won't be able to render the decoded MVC stream.
- Application may return error for some MJPEG streams decoded with option –low_latency.
- In case of using HEVC plugin (h265 video type), only software implementation of that plugin is used by default (even if you provide -hw option). To force usage of HEVC HW plugin implementation, please use -p option with proper plugin GUID.
- VP8 HW decoding is not working if system memory is used.
- SW HEVC plugin in 10bit mode cannot be used together with HW library VPP. Although library allows that, this is bad practice because additional per-pixel data shift is required. Please use HW HEVC + HW library or SW HEVC + SW library instead.
- Low latency mode (-low_latency) is not compatible with HEVC decoder.
- Sample may not function properly on systems that have a non-Intel VGA controller as the first (primary) because Intel device is not first in the list.

  To workaround this issue, swap names of DRI device files:

  ```
  $ cd /dev && mv card0 tmp && mv card1 card0 && mv tmp card1
  ```

  and do the same for the files `control64/65` and `renderD128/129`

Legal Information

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

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
