Encoding Sample

Overview

Encoding Sample works with **Intel® Media Server Studio 2018 - SDK for Linux* Server** (hereinafter referred to as "**SDK**").

It demonstrates how to use the **SDK** API to create a simple console application that performs preprocessing and encoding of an uncompressed video stream according to a specific video compression standard. Also the sample shows how to integrate user-defined functions for video processing (on example of picture rotation plug-in) into **SDK** encoding pipeline.

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

Encoding Sample supports the following video formats:

Format type	
input (uncompressed)	YUV420, NV12
output (compressed)	H.264 (AVC, MVC – Multi-View Coding), H.265 (with **HEVC**), MPEG-2 video, JPEG*/Motion JPEG, HEVC (High Efficiency Video Coding)

Note: For format YUV420, **Encoding Sample** assumes the order Y, U, V in the input file.

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`.

To enable V4L2 option during compilation, set --enable-v4l2=yes option while running build.pl

Running the Software

See `<install-folder>\Media_Samples_Guide.pdf`.

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

h264|h265|mpeg2|mvc|jpeg	Output video type. The use of option h265 is possible only if **HEVC** installed. The option –q is mandatory in case of JPEG encoding.
-i <InputFile>	Input (uncompressed) video file, name and path. In case of MVC -i option must be specified for each input YUV file. Only 2 views are supported
-o <Output>	Output (compressed) video file, name and path
-w <width>	Width of input video frame
-h <height>	Height of input video frame

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
-nv12	Signals that the input stream is in NV12 color format
-yuy2	Signals that the input stream is in YUY2, works for JPEG encode only
-rgb4	Signals that the input stream is in RGB4, works for JPEG encode only
-p010	Signals that the input stream is in P010, works for HEVC encode only
-ec::p010	Signals that an encoder should use P010 surfaces. If input has different FourCC vpp will be enabled. Works for HEVC encode only.
-tff|bff	Signals that the input stream is interlaced (top|bottom field first). If this option is not specified, progressive stream is expected.
-hw	Use platform-specific implementation of **SDK** (default)
-sw	Use software implementation of **SDK** (platform-specific implementation is used by default)
-b <bitrate>	Bitrate of the encoded stream in Kbits/second, supported for all encoders except JPEG*/ Motion JPEG
-f <framerate>	Frame rate of the encoded stream (30 by default)
-g <size>	GOP size (default 256)
-vbr	Variable bitrate control
-idr_interval <size>	IDR interval: default 0 means every I is an IDR, 1 means every other I frame is an IDR, etc...

-CodecProfile <profile>	Specifies codec profile
-CodecLevel <level>	Specifies codec level
-GopOptFlag:closed	Encoder generates closed GOP. Frames in this GOP do not use frames in previous GOP as reference.
-GopOptFlag:strict	Encoder strictly follows given GOP structure as defined by parameter GopPicSize, GopRefDist etc.
-InitialDelayInKB <size>	The HRD decoder starts decoding after the buffer reaches the initial size InitialDelayInKB, which is equivalent to reaching an initial delay of InitialDelayInKB*8000/TargetKbps ms
-BufferSizeInKB <size>	Represents the maximum possible size of any compressed frames
-MaxKbps <size>	For variable bitrate control, specifies the maximum bitrate at which the encoded data enters the Video Buffering Verifier buffer
-u <quality, speed, balanced>	Target usage (balanced by default). This parameter specifies a trade-off between quality and speed. Supported for all encoders except JPEG*/Motion JPEG
-q	The mandatory quality parameter for JPEG/Motion JPEG encoder (not valid for other encoders). In range [1,100], 100 is the best quality.
-la	Use the look ahead bitrate control algorithm (LA BRC) for H.264 encoder. Supported only with –hw library on processors with Intel® Iris™ Pro Graphics, Intel® Iris™ Graphics or Intel® HD Graphics 4200+ Series.
-lad	Depth parameter for the LA BRC, the number of frames to be analyzed before encoding. In range [10,100].
-cqp	Use constant quantization parameter (CQP BRC) bitrate control method (by default constant bitrate control method is used), should be used along with `-qpi, -qpp, -qpb`.
-qpi	Constant quantizer for I frames (if bitrace control method is CQP). In range [1,51]. 0 by default, i.e.no limitations on QP.
-qpp	Constant quantizer for P frames (if bitrace control method is CQP). In range [1,51]. 0 by default, i.e.no limitations on QP.
-qpb	Constant quantizer for B frames (if bitrace control method is CQP). In range [1,51]. 0 by default, i.e.no limitations on QP.
-qsv-ff	Enable QSV-FF mode
-num_slice	Number of slices in each video frame. 0 by default. If `num_slice` equals zero, the encoder may choose any slice partitioning allowed by the codec standard.
-mss	Maximum slice size in bytes. Supported only with harware library (`-hw`) and H.264 encoder. This option is not compatible with `-num_slice`.
-WeightedPred <default | implicit>	Weighted prediction mode.
-WeightedBiPred <default | implicit>	Weighted bi-prediction mode.

-dstw <width>	Width of encoded video frame. If specified and the value here is different from the value specified with `-w`, the encoder invokes video preprocessing (VPP) for scaling (resizing).
-dsth <height>	Height of encoded video frame. If specified and the value here is different from the value specified with –h, the encoder invokes video preprocessing (VPP) for scaling (resizing).
-angle 180	Invokes sample plug-in for 180 degrees picture rotation. CPU implementation is used by default. Rotate plugin module `sample_rotate_plugin.so` must be available when running the application with this option.
-opencl	Invokes Intel®OpenCL™ implementation of 180 degrees picture rotation. Rotate plugin module `libsample_plugin_opencl.so` must be available. File `ocl_rotate.cl` must exist in the local folder when running the application with this option.
-async	Depth of asynchronous pipeline. default value is 4. must be between 1 and 20.
-bref	Arrange B frames in B pyramid reference structure
-nobref	Do not use B-pyramid (by default the decision is made by library)
-gpucopy::<on,off>	Enable/disable GPU Copy functionality
-re	Enable region encode mode. Works only with HEVC encoder
-r <distance>	Distance between I- or P- key frames (1 means no B-frames)
-x <numRefs>	Number of reference frames. By default it is set to 1
-timeout	Encode in a loop not less than specific time in seconds. Performs complete input stream encoding on every iteration. Output file frames amount can be bigger than in input due to buffered frames in encoder. Output file is rewrote every iteration
-uncut	Do not cut output file in looped mode (in case of -timeout option)
-membuf	Read specific amount of frames into memory and encode them in a loop during the time specified in -timeout option. Can be used as encoder benchmark
-gpb:<on,off>	GPB control. Turn this option OFF to make HEVC encoder use regular P-frames instead of GPB.
-n <frames number>	Tumber of frames to process
-mfs <maxFrameSize>	Maximum frame size in bytes. Supported only with h264 and hevc codec for VBR mode.
-dump	Dump MSDK components configuration to the file in text form.
-signal:<mode>	Represents transfer matrix coefficients for mfxExtVideoSignalInfo. 0 - unknown, 1 - BT709, 2 - BT601
-ppyr:<on,off>	Turn this option ON to enable P-pyramid (by default the decision is made by library)
-LowDelayBRC	Strictly obey average frame size set by MaxKbps
-ir_type <type>	Intra refresh type. 0 - no refresh, 1 - vertical refresh, 2 - horisontal refresh, 3 - slice refresh
-WeightedPred:default|implicit	Weighted prediction mode: default, implicit

-WeightedBiPred:default|implicit	Weighted B prediction mode: default, implicit
-ir_cycle_size <size>	Number of pictures within refresh cycle starting from 2
-ir_qp_delta <delta>	QP difference for inserted intra MBs. This is signed value in [-51, 51] range
-ir_cycle_dist <dist>	Distance between the beginnings of the intra-refresh cycles in frames
-usei	Insert user data unregistered SEI. eg: 7fc92488825d11e7bb31be2e44b06b34:0:MSDK (uuid:type<0-preifx/1-suffix>:message) the suffix SEI for HEVCe can be inserted when CQP used or HRD disabled
-extbrc:<on,off>	External BRC for AVC and HEVC encoders
-?	Print help

If V4L2 support is enabled during compilation, additional options are available:

-d	Device video node (eg: /dev/video0)
-p	Mipi Port number (eg: Port 0)
-m	Mipi Mode Configuration [PREVIEW/CONTINUOUS/STILL/VIDEO]
-uyvy	Input Raw format types V4L2 Encode
-YUY2	Input Raw format types V4L2 Encode
-i::v4l2	Enable v4l2 mode

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

```
$ sample_encode h264 -i input.yuv -o output.h264 –w 720 –h 480 –b 10000 –f
 30 –u quality –d3d –hw
$ sample_encode mpeg2 -i input.yuv -o output.mpeg2 –w 1920 –h 1080 –b 15000
 -u speed –nv12 –tff -hw
$ sample_encode h264 -i input.yuv -o output.h264 –w 1920 –h 1080 –dstw 360 –
dsth 240 –b 1000 –u balanced -hw
```

Note 1: You need to have **HEVC** installed to run with h265 codec. In case of HW library it will firstly try to load HW HEVC plugin in case of failure - it will try SW one if available.

Tip:

To achieve better performance, use input streams in NV12 color format. If the input stream is in YUV420 format, each frame is converted to NV12 which reduces overall performance.

HEVC Encode Plugins

HEVC codec is implemented as a plugin unlike codecs such as MPEG2 and AVC. There are 3 implementations of HEVC encoder: Hardware (HW), Software (SW) and GPU-Accelerated (GACC) plugins.

Note 1: The HEVC SW and GACC plugins are available only 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 plugin for HEVC encode is supported starting from 6th Generation of Intel CoreTM Processors, Intel® Xeon® E3-1200 and E3-1500 v5 Family with Intel® Processor Graphics 500 Series (codename Skylake).

Note 3: Encoding sample loads the HW HEVC encode plugin with HW library and SW encode with SW library by default. You can enforce a plugin to be loaded by specifying the plugin ID using "-p" parameter and hexadecimal GUID.

Examples of running HW HEVC encoder with HW library:

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

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

Different ways to run SW HEVC encoder (the first example uses SW library, the second one uses HW library):

```
$ sample_encode h265 -i input.yuv -o output.h265 –w 720 –h 480 –b 10000 –f
 30 –u quality -sw
$ sample_encode h265 -i input.yuv -o output.h265 –w 720 –h 480 –b 10000 –f
 30 –u quality -p 2fca99749fdb49aeb121a5b63ef568f7
```

To run the HEVC GACC encoder, you have to specify the “-p” parameter with GUID. Please note that GACC plugin works only with HW library.

The following command-lines will force sample to use the HEVC GACC Encode plugin:

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

To run HEVC plugins in 10 bit mode, you have to specify the “-ec::p010” or "-p010" option:

```
$ sample_encode h265 -i input.yuv -o output.h265 -w 720 -h 480 -b 10000 -
ec::p010
```

```
$ sample_encode h265 -i input.p010 -o output.h265 -w 720 -h 480 -b 10000 -
p010
```

Known Limitations

- Not all combinations of optional switches are supported. If the option `–angle 180` or `–opencl` is specified, options `–tff|bff, -dstw, -dsth, -d3d` and MVC output are not available.
- **Encoding Sample** if run with `–opencl` option requires input video frame width to be aligned by 4.
- In case of using HEVC plugin (h265 video type), plugin type (hardware or software) used by default is set depending on -sw or -hw sample options. However, hardware HEVC plugins work on specific platforms only. To force usage of specific HEVC plugin implementation, please use -p option with proper plugin GUID.
- 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`
- HECV 10Bit encoding works with HEVC SW plugin only due to library limitations

Legal Information