Tutorials Command Line Reference

Intel® Media SDK Tutorials

Intel Corporation

Table of Contents

1 General considerations

Intel Media SDK Tutorials package was designed in the simplistic manner. Each tutorial suites the purpose to demonstrate some technique of working with the Media SDK library. For the simplicity sample code supports only minimum required number of options without which it is impossible to variate input data and produce correct result. This command line reference is divided into few parts. Each part describes group of tutorials which have the same or similar list of command line options and arguments. The parts are:

- **Basic tutorials**.
- **Decoding tutorials**. The only configurable component in these tutorials is decoder (H.264). Configuration parameters are input and output streams.
- **Encoding tutorials**. The only configurable component is encoder (H.264). Configuration parameters (besides input and output stream) - width, height, bitrate, framerate.
- **VPP tutorials**. The only configurable component is VPP. Configuration parameters (besides input and output stream) - input stream width and height.
- **Transcoding tutorials**. Configurable components are decoder (H.264) and encoder (H.264). Configuration parameters are: input/output streams, bitrate/framerate to encode.

2 Basic tutorials

2.1 simple_1_session

`simple_session` *[-sw|-hw|-auto]* [Command]

Tutorial initializes Intel Media SDK session of a specified type. If `-auto` is specified (the default) Media SDK dispatcher will automatically select library implementation to use. If `-hw` is specified HW implementation will be used, `-sw` - SW one. Depending on Media SDK distribution different implementations may present. Options to adjust session type are applicable for all other tutorials descried in this reference.

`-sw` [Option]

Loads SW Media SDK Library implementation

`-hw` [Option]

Loads HW Media SDK Library implementation

`-auto` [Option]

Automatically choses Media SDK library implementation

3 Decoding tutorials

3.1 simple_2_decode

`simple_decode` *[-sw|-hw|-auto] input-file* [*output-file*] [Command]
Tutorial demonstrates how to decode given raw video stream (`input-file`) of H.264 format. If `output-file` was specified, decoded YUV will be written into it. If the `output-file` is omitted, tutorial can be used to estimate constructed pipeline's performance. Decoder is configured to produce decoded data in the system memory.

`-sw` [Option]
Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`input-file` [Argument]
Mandatory argument. Sets incoming input bitstream to process. Input data should be in raw H.264 format.

`output-file` [Argument]
Optional argument. Sets uncompressed video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

3.2 simple_2_decode_vmem

`simple_decode_vmem` *[-sw|-hw|-auto] input-file* [*output-file*] [Command]
Tutorial demonstrates how to decode given raw video stream (`input-file`) of H.264 format. If `output-file` was specified, decoded YUV will be written into it. If the `output-file` is omitted, tutorial can be used to estimate constructed pipeline's performance. Decoder is configured to produce decoded data in the video memory.

`-sw` [Option]
Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`input-file` [Argument]
Mandatory argument. Sets incoming input bitstream to process. Input data should be in raw H.264 format.

`output-file` [Argument]
Optional argument. Sets uncompressed video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

3.3 simple_6_decode_vpp_postproc

`simple_decode_vpp_pp` *[-sw|-hw|-auto] input-file* [*output-file*] [Command]

Tutorial demonstrates how to decode given raw video stream (`input-file`) of H.264 format and perform some operation with the decoded frames thru the Video Post-Processing (VPP) component. If `output-file` was specified, decoded YUV will be written into it. If the `output-file` is omitted, tutorial can be used to estimate constructed pipeline's performance. The performed VPP operation is x2 downscaling.

`-sw` [Option]

Loads SW Media SDK Library implementation

`-hw` [Option]

Loads HW Media SDK Library implementation

`-auto` [Option]

Automatically choses Media SDK library implementation

`input-file` [Argument]

Mandatory argument. Sets incoming input bitstream to process. Input data should be in raw H.264 format.

`output-file` [Argument]

Optional argument. Sets uncompressed video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

4 Encoding tutorials

4.1 simple_3_encode

`simple_encode` *[-sw|-hw|-auto] -g WxH -b bitrate -f framerate* [*input-file*] [*output-file*] [Command]

Tutorial demonstrates how to encode given YUV video stream (`input-file`) in H.264 format. If `output-file` was specified, encoded bitstream will be written into it. If the `input-file` and `output-file` are omitted, tutorial can be used to estimate constructed pipeline's performance (to some degree). Encoder is configured to receive input data from system memory.

`-sw` [Option]

Loads SW Media SDK Library implementation

`-hw` [Option]

Loads HW Media SDK Library implementation

`-auto` [Option]

Automatically choses Media SDK library implementation

`-g` *WxH* [Option]

Mandatory option. Sets input video geometry, i.e. width and height. Example: `-g 1920x1080`.

`-b` *bitrate* [Option]

Mandatory option. Sets bitrate with which data should be encoded, in KBits-per-second. Example: `-b 5000` to encode data at 5Mbit.

`-f` *framerate* [Option]

Mandatory option. Sets framerate with which data should be encoded in the form `-f nominator/denominator`. Example: `-f 30/1`.

`input-file` [Argument]

Optional argument. Sets incoming input file containing uncompressed video. If file will not be specified, tutorial will work in the performance mode: input will be simulated by producing empty input frames filled with some color.

`output-file` [Argument]

Optional argument. Sets raw H.264 video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

4.2 simple_3_encode_vmem

`simple_encode_vmem` *[-sw|-hw|-auto] -g WxH -b bitrate -f framerate* [*input-file*] [*output-file*] [Command]

Tutorial demonstrates how to encode given YUV video stream (`input-file`) in H.264 format. If `output-file` was specified, encoded bitstream will be written into it. If the `input-file` and `output-file` are omitted, tutorial can be used to estimate constructed pipeline's performance (to some degree). Encoder is configured to receive input data from video memory.

`-sw` [Option]

Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`-g` *WxH* [Option]
Mandatory option. Sets input video geometry, i.e. width and height. Example: `-g 1920x1080`.

`-b` *bitrate* [Option]
Mandatory option. Sets bitrate with which data should be encoded, in KBits-per-second. Example: `-b 5000` to encode data at 5Mbit.

`-f` *framerate* [Option]
Mandatory option. Sets framerate with which data should be encoded in the form `-f nominator/denominator`. Example: `-f 30/1`.

`input-file` [Argument]
Optional argument. Sets incoming input file containing uncompressed video. If file will not be specified, tutorial will work in the performance mode: input will be simulated by producing empty input frames filled with some color.

`output-file` [Argument]
Optional argument. Sets raw H.264 video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

4.3 simple_3_encode_vmem_async

`simple_encode_vmem_async` *[-sw|-hw|-auto] -g WxH -b bitrate -f framerate* [*input-file*] [*output-file*] [Command]
Tutorial demonstrates how to encode given YUV video stream (`input-file`) in H.264 format. If `output-file` was specified, encoded bitstream will be written into it. If the `input-file` and `output-file` are omitted, tutorial can be used to estimate constructed pipeline's performance (to some degree). Encoder is configured to receive input data from system memory. Constructed pipeline is asynchronious and should be faster than pipelines in simple_3_encode and simple_3_encode_vmem samples.

`-sw` [Option]
Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`-g` *WxH* [Option]
Mandatory option. Sets input video geometry, i.e. width and height. Example: `-g 1920x1080`.

`-b` *bitrate* [Option]
Mandatory option. Sets bitrate with which data should be encoded, in KBits-per-second. Example: `-b 5000` to encode data at 5Mbit.

`-f` *framerate* [Option]
Mandatory option. Sets framerate with which data should be encoded in the form `-f nominator/denominator`. Example: `-f 30/1`.

`input-file` [Argument]
Optional argument. Sets incoming input file containing uncompressed video. If file will not be specified, tutorial will work in the performance mode: input will be simulated by producing empty input frames filled with some color.

`output-file` [Argument]
Optional argument. Sets raw H.264 video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

4.4 simple_6_encode_vmem_lowlatency

`simple_encode_vmem_lowlat` *[-sw|-hw|-auto] -g WxH -b bitrate -f framerate [–measure-latency|–no-measure-latency]* [*input-file*] [*output-file*] [Command]
Tutorial demonstrates how to encode given YUV video stream (`input-file`) in H.264 format. If `output-file` was specified, encoded bitstream will be written into it. If the `input-file` and `output-file` are omitted, tutorial can be used to estimate constructed pipeline's performance (to some degree). Encoder is configured to work in low latency mode, producing encoded data as fast as possible.

`-sw` [Option]
Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`-g` *WxH* [Option]
Mandatory option. Sets input video geometry, i.e. width and height. Example: `-g 1920x1080`.

`-b` *bitrate* [Option]
Mandatory option. Sets bitrate with which data should be encoded, in KBits-per-second. Example: `-b 5000` to encode data at 5Mbit.

`-f` *framerate* [Option]
Mandatory option. Sets framerate with which data should be encoded in the form `-f nominator/denominator`. Example: `-f 30/1`.

`--measure-latency` [Option]
Optional argument. With the option set tutorial calculates and prints latency statistics. This is the default.

`--no-measure-latency` [Option]
Optional argument. With the option set tutorial will not calculate and print latency statistics.

`input-file` [Argument]
Optional argument. Sets incoming input file containing uncompressed video. If file will not be specified, tutorial will work in the performance mode: input will be simulated by producing empty input frames filled with some color.

`output-file` [Argument]
Optional argument. Sets raw H.264 video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

4.5 simple_6_encode_vmem_vpp_preproc

`simple_encode_vmem_preproc` *[-sw|-hw|-auto] -g WxH -b bitrate -f framerate* [*input-file*] [*output-file*] [Command]
Tutorial demonstrates how to encode given RGB video stream (`input-file`) in H.264 format after applying some Video Pre-Processing (VPP) operation. If `output-file` was specified, encoded bitstream will be written into it. If the `input-file` and `output-file` are omitted, tutorial can be used to estimate constructed pipeline's performance (to some degree). The performed VPP operation is RGB to NV12 color space conversion.

`-sw` [Option]
Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`-g` *WxH* [Option]
Mandatory option. Sets input video geometry, i.e. width and height. Example: `-g 1920x1080`.

`-b` *bitrate* [Option]
Mandatory option. Sets bitrate with which data should be encoded, in KBits-per-second. Example: `-b 5000` to encode data at 5Mbit.

`-f` *framerate* [Option]
Mandatory option. Sets framerate with which data should be encoded in the form `-f nominator/denominator`. Example: `-f 30/1`.

`input-file` [Argument]
Optional argument. Sets incoming input file containing uncompressed video. If file will not be specified, tutorial will work in the performance mode: input will be simulated by producing empty input frames filled with some color.

`output-file` [Argument]
Optional argument. Sets raw H.264 video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

5 Video PreProcessing (VPP) tutorials

5.1 simple_4_vpp_resize_denoise

`simple_vpp` *[-sw|-hw|-auto] -g WxH* [*input-file*] [*output-file*] [Command]
Tutorial demonstrates how to run some Video Pre/Post-Processing (VPP) operation in the given input YUV stream (`input-file`). If `output-file` was specified, produced stream will be written into it. If the `input-file` and `output-file` are omitted, tutorial can be used to estimate constructed pipeline's performance. Constructed pipeline is configured to work on system memory on the input and output. The performed VPP operation is x2 downscaling and noise reduction.

`-sw` [Option]
Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`-g` *WxH* [Option]
Mandatory option. Sets input video geometry, i.e. width and height. Example: `-g 1920x1080`.

`input-file` [Argument]
Optional argument. Sets incoming input file containing uncompressed video. If file will not be specified, tutorial will work in the performance mode: input will be simulated by producing empty input frames filled with some color.

`output-file` [Argument]
Optional argument. Sets uncompressed video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

5.2 simple_4_vpp_resize_denoise_vmem

`simple_vpp_vmem` *[-sw|-hw|-auto] -g WxH* [*input-file*] [*output-file*] [Command]
Tutorial demonstrates how to run some Video Pre/Post-Processing (VPP) operation in the given input YUV stream (`input-file`). If `output-file` was specified, produced stream will be written into it. If the `input-file` and `output-file` are omitted, tutorial can be used to estimate constructed pipeline's performance. Constructed pipeline is configured to work on video memory on the input and output. The performed VPP operation is x2 downscaling and noise reduction.

`-sw` [Option]
Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`-g` *WxH* [Option]
Mandatory option. Sets input video geometry, i.e. width and height. Example: `-g 1920x1080`.

`input-file` [Argument]
Optional argument. Sets incoming input file containing uncompressed video. If file will not be specified, tutorial will work in the performance mode: input will be simulated by producing empty input frames filled with some color.

`output-file` [Argument]
Optional argument. Sets uncompressed video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

6 Transcoding tutorials

6.1 simple_5_transcode

`simple_transcode` *[-sw|-hw|-auto] -b bitrate -f framerate input-file* [*output-file*] [Command]

Tutorial demonstrates how to transcode given input raw H.264 video stream (`input-file`) into H.264 stream with different parameters. If `output-file` was specified, produced stream will be written into it. If `output-file` is omitted, tutorial can be used to estimate constructed pipeline's performance. Constructed pipeline is configured to share system memory between decoder and encoder.

`-sw` [Option]
Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`-b` *bitrate* [Option]
Mandatory option. Sets bitrate with which data should be encoded, in KBits-per-second. Example: `-b 5000` to encode data at 5Mbit.

`-f` *framerate* [Option]
Mandatory option. Sets framerate with which data should be encoded in the form `-f nominator/denominator`. Example: `-f 30/1`.

`input-file` [Argument]
Mandatory argument. Sets incoming input bitstream to process. Input data should be in raw H.264 format.

`output-file` [Argument]
Optional argument. Sets raw H.264 video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

6.2 simple_5_transcode_vmem

`simple_transcode_vmem` *[-sw|-hw|-auto] -b bitrate -f framerate input-file* [*output-file*] [Command]

Tutorial demonstrates how to transcode given input raw H.264 video stream (`input-file`) into H.264 stream with different parameters. If `output-file` was specified, produced stream will be written into it. If `output-file` is omitted, tutorial can be used to estimate constructed pipeline's performance. Constructed pipeline is configured to share video memory between decoder and encoder.

`-sw` [Option]
Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`-b` *bitrate* [Option]
Mandatory option. Sets bitrate with which data should be encoded, in KBits-per-second. Example: `-b 5000` to encode data at 5Mbit.

`-f` *framerate* [Option]
Mandatory option. Sets framerate with which data should be encoded in the form `-f nominator/denominator`. Example: `-f 30/1`.

`input-file` [Argument]
Mandatory argument. Sets incoming input bitstream to process. Input data should be in raw H.264 format.

`output-file` [Argument]
Optional argument. Sets raw H.264 video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

6.3 simple_5_transcode_opaque

`simple_transcode_opaque` *[-sw|-hw|-auto] -b bitrate -f framerate input-file* [*output-file*] [Command]
Tutorial demonstrates how to transcode given input raw H.264 video stream (`input-file`) into H.264 stream with different parameters. If `output-file` was specified, produced stream will be written into it. If `output-file` is omitted, tutorial can be used to estimate constructed pipeline's performance. Constructed pipeline is configured to share opaque memory between decoder and encoder.

`-sw` [Option]
Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`-b` *bitrate* [Option]
Mandatory option. Sets bitrate with which data should be encoded, in KBits-per-second. Example: `-b 5000` to encode data at 5Mbit.

`-f` *framerate* [Option]
Mandatory option. Sets framerate with which data should be encoded in the form `-f nominator/denominator`. Example: `-f 30/1`.

`input-file` [Argument]
Mandatory argument. Sets incoming input bitstream to process. Input data should be in raw H.264 format.

`output-file` [Argument]
Optional argument. Sets raw H.264 video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

6.4 simple_5_transcode_opaque_async

`simple_transcode_opaque_async` *[-sw|-hw|-auto] -b bitrate -f framerate input-file* [*output-file*] [Command]

Tutorial demonstrates how to transcode given input raw H.264 video stream (`input-file`) into H.264 stream with different parameters. If `output-file` was specified, produced stream will be written into it. If `output-file` is omitted, tutorial can be used to estimate constructed pipeline's performance. Constructed pipeline is configured to share opaque memory between decoder and encoder and work in the asynchronious mode.

`-sw` [Option]

Loads SW Media SDK Library implementation

`-hw` [Option]

Loads HW Media SDK Library implementation

`-auto` [Option]

Automatically choses Media SDK library implementation

`-b` *bitrate* [Option]

Mandatory option. Sets bitrate with which data should be encoded, in KBits-per-second. Example: `-b 5000` to encode data at 5Mbit.

`-f` *framerate* [Option]

Mandatory option. Sets framerate with which data should be encoded in the form `-f nominator/denominator`. Example: `-f 30/1`.

`input-file` [Argument]

Mandatory argument. Sets incoming input bitstream to process. Input data should be in raw H.264 format.

`output-file` [Argument]

Optional argument. Sets raw H.264 video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

6.5 simple_6_transcode_opaque_lowlatency

`simple_transcode_opaque_lowlat` *[-sw|-hw|-auto] -b bitrate -f framerate input-file* [*output-file*] [Command]

Tutorial demonstrates how to transcode given input raw H.264 video stream (`input-file`) into H.264 stream with different parameters. If `output-file` was specified, produced stream will be written into it. If `output-file` is omitted, tutorial can be used to estimate constructed pipeline's performance. Constructed pipeline is configured to share opaque memory between decoder and encoder and work in the low latancy mode, i.e. produce routput as fast as possible.

`-sw` [Option]

Loads SW Media SDK Library implementation

`-hw` [Option]

Loads HW Media SDK Library implementation

`-auto` [Option]

Automatically choses Media SDK library implementation

`-b` *bitrate* [Option]
Mandatory option. Sets bitrate with which data should be encoded, in KBits-per-second. Example: `-b 5000` to encode data at 5Mbit.

`-f` *framerate* [Option]
Mandatory option. Sets framerate with which data should be encoded in the form `-f nominator/denominator`. Example: `-f 30/1`.

`input-file` [Argument]
Mandatory argument. Sets incoming input bitstream to process. Input data should be in raw H.264 format.

`output-file` [Argument]
Optional argument. Sets raw H.264 video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.

6.6 simple_5_transcode_opaque_async_vppresize

`simple_transcode_opaque_async_vppresize` *[-sw|-hw|-auto] -b bitrate -f framerate input-file* [*output-file*] [Command]
Tutorial demonstrates how to transcode given input raw H.264 video stream (`input-file`) into H.264 stream with different parameters. This tutorial adds VPP processing (x4 downscaling) in the pipeline. If `output-file` was specified, produced stream will be written into it. If `output-file` is omitted, tutorial can be used to estimate constructed pipeline's performance. Constructed pipeline is configured to share opaque memory between decoder and encoder and work in the asynchronious mode.

`-sw` [Option]
Loads SW Media SDK Library implementation

`-hw` [Option]
Loads HW Media SDK Library implementation

`-auto` [Option]
Automatically choses Media SDK library implementation

`-b` *bitrate* [Option]
Mandatory option. Sets bitrate with which data should be encoded, in KBits-per-second. Example: `-b 5000` to encode data at 5Mbit.

`-f` *framerate* [Option]
Mandatory option. Sets framerate with which data should be encoded in the form `-f nominator/denominator`. Example: `-f 30/1`.

`input-file` [Argument]
Mandatory argument. Sets incoming input bitstream to process. Input data should be in raw H.264 format.

`output-file` [Argument]
Optional argument. Sets raw H.264 video file to write output data. If file will not be specified, tutorial will work in the perfromance mode: it will process data, but will not produce any output.