Scalable Video Technology for HEVC (SVT-HEVC) Software Encoder

User Guide

© Intel Corporation

Revision History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Comments** |
| 0.1 | October, 20 2017 | Initial draft |
| 0.2 | January, 11 2018 | Add new configuration parameters |
| 0.3 | March, 23, 2018 | Add new configuration parameters |
| 0.4 | March 27, 2018 | Added Source Build instructions and config parameters |

Table of Contents

[1. Introduction 4](#_Toc510031433)

[2. System Requirements 4](#_Toc510031434)

[2.1 CPU requirements 4](#_Toc510031435)

[2.2 RAM requirements 4](#_Toc510031436)

[2.3 Operating systems 4](#_Toc510031437)

[Windows\* Operating Systems (64-bit): 4](#_Toc510031438)

[Linux\* Operating Systems (64-bit): 4](#_Toc510031439)

[2.4 Build the code 5](#_Toc510031440)

[Windows\* Operating Systems (64-bit): 5](#_Toc510031441)

[Linux\* Operating Systems (64-bit): 5](#_Toc510031442)

[2.5 Installation 5](#_Toc510031443)

[3 User Guide 6](#_Toc510031444)

[3.1 Input Video Format 6](#_Toc510031445)

[3.2 Compressed 10bit format 6](#_Toc510031446)

[3.2.1 Unpack the 10 bit picture 6](#_Toc510031447)

[3.2.2 Compress the 2 bit Plane 7](#_Toc510031448)

[3.2.3 Unroll the 64x64 7](#_Toc510031449)

[3.3 Running the encoder 8](#_Toc510031450)

[2.1.1 List of all configuration parameters 9](#_Toc510031451)

[4 Best Known Configurations (BKC) 15](#_Toc510031452)

[4.1 Hardware BKC: 15](#_Toc510031453)

[4.2 Software BKC: 15](#_Toc510031454)

[10 bit Input YUV: 15](#_Toc510031455)

[Windows\* OS (Tested on Windows\* Server 2016) 15](#_Toc510031456)

[Linux\* OS (Tested on Ubuntu\* Server 16.04) 15](#_Toc510031457)

[Legal Disclaimer 16](#_Toc510031458)

# Introduction

This document describes the system requirements and how to use the Scalable Video Technology for HEVC (SVT-HEVC) software encoder. In particular, this user guide section describes how to run the sample application with the respective dynamically linked library.

# System Requirements

The SVT-HEVC encoder library was developed to be supported on x86 for Windows\* and Linux\* operating systems.

## CPU requirements

In order to achieve the performance targeted by the deliverable, the specific CPU model listed in the release notes would need to be used when running the encoder. Otherwise, the encoder should run on any 5th Generation Intel Core™ Processors (formerly Broadwell) CPUs or newer producing the same output.

## RAM requirements

In order to run the highest resolution supported by the encoder, at least 16GB of RAM is required to run a single 4kp60 10-bit encode. The encoder application will display an error if the system does not have enough RAM to support such. The following table show the minimum amount of RAM required for some standard resolutions of 10bit video per channel:

|  |  |
| --- | --- |
| **Resolution** | **Minimum Footprint in GB** |
| 4k | 16 |
| 1080p | 6 |
| 720p/1080i | 4 |
| 480p | 2.5 |

## Operating systems

The list below includes the operating systems that the encoder application and library could run on, assuming the above pre-requisites are met.

### Windows\* Operating Systems (64-bit):

* + - Windows\* 7
    - Windows\* 10
    - Windows\* Server 2016 std

### Linux\* Operating Systems (64-bit):

* + - Ubuntu\* 16.04
    - Ubuntu\* 16.04 Server

## Build the code

The list below includes the build tools necessary for the encoder application and library to build properly.

### Windows\* Operating Systems (64-bit):

* Build requirements
  + - Visual Studio\* 2015
    - YASM Assembler version 1.2.0 or later
  + Download the yasm package from the following [link](http://www.tortall.net/projects/yasm/releases/vsyasm-1.2.0-win64.zip)
  + Copy the Yasm files [all 4 of them] under the bin folder of your Visual Studio\* installation folder e.g.:
    - C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\bin
* Build instructions
  + - Open the “Build\VS\ebHevcEnc.sln” file and build the solution
    - Or run the “Build\windows\build.bat” script
* Binaries Location:
  + Binaries can be found under Bin/Release and / or Bin/Debug

### Linux\* Operating Systems (64-bit):

* Build requirements
  + GCC 5.4.0
  + CMake 3.5.1
  + YASM Assembler version 1.2.0 or later
* Build instructions
  + cd Build/linux
  + chmod +x build.sh
  + ./build.sh <release | debug> (if none specified, both release and debug will be built)
* Binaries Location:
  + Binaries can be found under Bin/Release and / or Bin/Debug

## Installation

For the binaries to operate properly on your system, the following conditions have to be met:

* Windows\*:
  + On any of the Windows\* operating systems listed in section 2.3, Install Visual Studio 2015
  + Once the installation is complete, copy the binaries in a location making sure that both the sample application “ebHevcEncApp.exe” and library “ebHevcEncLib.dll” are in the same folder.
  + Open the command line at the chosen location and run the sample application to encode.
* Linux\*:
  + On any of the Linux\* operating systems listed in section 2.3, copy the binaries under a location of your choice.
  + Change the permissions on the sample application “HevcEncoderApp” executable by running the command:
    - chmod +x HevcEncoderApp
  + Open the terminal and cd into your chosen location
  + Run the sample application to encode.

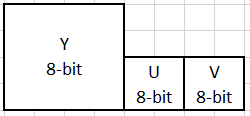
# User Guide

This section describes how to run the sample encoder application that uses the SVT-HEVC encoder library. It describes the command line input parameters, input video format, and the resulting outputs.

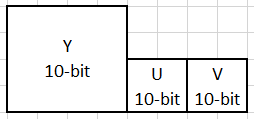
## Input Video Format

The SVT-HEVC encoder supports the following input formats:

* 8-bit yuv420p



* 10bit yuv420p10le



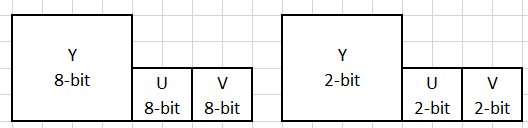
## Compressed 10bit format

In order to reduce the size of the input original YUV file, the SVT-HEVC encoder uses a compressed 10bit format allowing the software to achieve a higher density level. The conversion between the 10-bit yuv420p10le and the compressed 10-bit format is a lossless operation and is performed using the following steps:

### Unpack the 10 bit picture

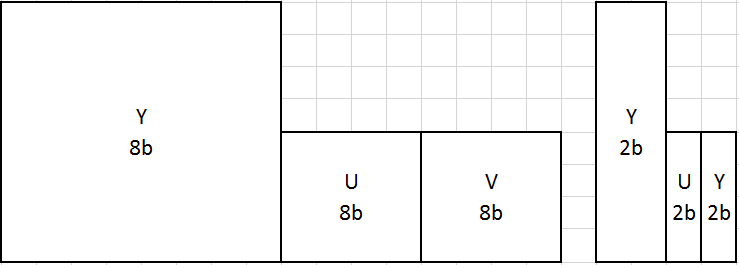
This step consists of separating the 10 bit video samples into 8 bit and 2 bit planes so that each 10bit picture will be represented as two separate pictures as shown in the figure below. As a result of the operation, the 2 least significant bits of the 10 will be written into a full byte.

10bit yuv420p10le unpacked



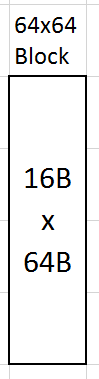
### Compress the 2 bit Plane

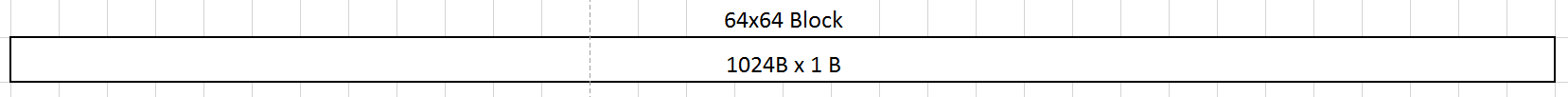
After unpacking the 2 bits from the 10 bit samples, they should be now compressed so that each 4x2bit samples are written into one byte. As a result, each 10bit picture will be represented as two separate pictures as shown in the figure below.



### Unroll the 64x64

Now for a faster read of the samples, the every 64x64 block of the 2 bit picture should be written into a one dimensional array. Therefore, the top left 64x64 sample block which is now written into a 16 bytes x 64 bytes after the compression of the 2bit samples, will be written into a 1024 bytes x 1 byte array as shown in the picture below.





64x64 block after unrolling

64x64 block after 2 bit compression

## Running the encoder

This section describes how to run the sample encoder application ebHevcEncApp.exe from the command line, including descriptions of the most commonly used input parameters and outputs.

The sample application typically takes the following command line parameters:

-c filename [**Optional**]

A text file that contains encoder parameters such as input file name, quantization parameter etc. Refer to the comments in the Config/Sample.cfg for specific details. The list of encoder parameters are also listed below. Note that command line parameters take precedence over the parameters included in the configuration file when there is a conflict.

-i filename **[Required]**

A YUV file (e.g. 8 bit 4:2:0 planar) containing the video sequence that will be encoded. The dimensions of each image are specified by –w and –h as indicated below.

-b filename **[Optional]**

The resulting encoded bit stream file in binary format. If none specified, no output bit stream will be produced by the encoder.

-w integer **[Required]**

The width of each input image in units of picture luma pixels. E.g. 1920

-h integer **[Required]**]

The height of each input image in units of picture luma pixels. E.g. 1080

-n integer **[Optional]**

The number of frames of the sequence to encode. E.g. 100. If the input frame count is larger than the number of frames in the input video, the encoder will loopback to the first frame when it’s done

-intra-period integer **[Optional]**

The intra period defines the interval of frames after which you insert an Intra refresh. It is strongly recommended to use (multiple of 8) -1 the closest to 1 second (e.g. 55, 47, 31, 23 should be used for 60, 50, 30, (24 or 25) respectively)

-rc integer **[Optional]**

This token sets the bitrate control encoding mode [1: Variable Bitrate, 0: Constant QP]. When rc is set to 1, it’s best to match the –lad (lookahead distance described in the next section) parameter to the -intra-period.

-speed-ctrl integer **[Optional]**

This token sets the encoder to automatically choose the best quality encoding mode that allows the encoder to run at a real-time speed set by the –fps parameter (described in the next section).

For example, the following command encodes 100 frames of the YUV video sequence into the bin bit stream file. The picture is 1920 luma pixels wide and 1080 pixels high using the Sample.cfg configuration. The QP equals 30 and the md5 checksum is not included in the bit stream.

> ebHevcEncApp.exe -c Sample.cfg -i CrowdRun\_1920x1080.yuv -w 1920 -h 1080 -n 100 -q 30 -intra-period 31 -b CrowdRun\_1920x1080\_qp30.bin

It should be noted that not all the encoder parameters present in the Sample.cfg can be changed using the command line.

### List of all configuration parameters

The encoder parameters present in the Sample.cfg file are listed in this table below along with their status of support, command line parameter and the range of values that the parameters can take.

**Note**: If the default value is “None”, the parameter is required to start encoding.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Encoder Parameter as shown in the configuration file** | **Command Line parameter** | **Range** | **Default** | **Description** |
|  | -nch | [1 - 6] | 1 | Number of encode instances |
|  | -c | any string | null | Configuration file path |
| **InputFile** | -i | any string | None | Input file path |
| **StreamFile** | -b | any string | null | output bitstream file path |
| **ErrorFile** | -errlog | any string | stderr | error log displaying configuration or encode errors |
| **UseQpFile** | -use-q-file | [0 - 1] | 0 | When set to 1, overwrite the picture qp assignment using qp values in QpFile |
| **QpFile** | -qp-file | any string | Null | Path to qp file |
| **EncoderMode** | -encMode | [1 - 6] | 2 | A preset defining the quality vs density tradeoff point that the encoding is to be performed at. (e.g. 1 is the highest quality mode, 6 is the highest density mode) |
| **LatencyMode** | -latency-mode | [0 - 1] | 0 | For lower latency (0: Normal Latency, 1: Low Latency) |
| **EncoderBitDepth** | -bit-depth | [8 , 10] | 8 | specifies the bit depth of the input video |
| **CompressedTenBitFormat** | -compressed-ten-bit-format | [0 - 1] | 0 | Offline packing of the 2bits: requires two bits packed input (0: OFF, 1: ON) |
| **SourceWidth** | -w | [64 - 4096] | None | Input source width |
| **SourceHeight** | -h | [0 - 2304] | None | Input source height |
| **FrameToBeEncoded** | -n | [0 - 2^31 -1] | 0 | Number of frames to be encoded, if number of frames is > number of frames in file, the encoder will loop to the beginning and continue the encode. Use -1 to not buffer. |
| **BufferedInput** | -nb | [-1, 1 to 2^31 -1] | -1 | number of frames to preload to the RAM before the start of the encode If -nb = 100 and –n 1000 -- > the encoder will encode the first 100 frames of the video 10 times |
| **Profile** | -profile | [1 - 2] | 2 | 1: Main, 2: Main 10 |
| **Tier** | -tier | [0 - 1] | 0 | 0: Main, 1: High |
| **Level** | -level | [1, 2, 2.1,3, 3.1, 4, 4.1, 5, 5.1, 5.2, 6, 6.1, 6.2] | 0 | 0 to 6.2 [0 for auto determine Level] |
| **FrameRate** | -fps | [0 - 2^64 -1] | 25 | If the number is less than 1000, the input frame rate is an integer number between 1 and 60, else the input number is in Q16 format (shifted by 16 bits) [Max allowed is 240 fps] |
| **FrameRateNumerator** | -fps-num | [0 - 2^64 -1] | 0 | Frame rate numerator e.g. 6000 |
| **FrameRateDenominator** | -fps-denom | [0 - 2^64 -1] | 0 | Frame rate denominator e.g. 100 |
| **Injector** | -inj | [0 - 1] | 0 | Enable injection of input frames at the specified framerate (0: OFF, 1: ON) |
| **InjectorFrameRate** | -inj-frm-rt | [1 - 240] | 60 | Frame Rate used for the injector. Recommended to match the encoder speed. |
| **SpeedControlFlag** | -speed-ctrl | [0 - 1] | 0 | Enables the Speed Control functionality to achieve the real-time encoding speed defined by –fps. When this parameter is set to 1 it forces –inj to be 1  -inj-frm-rt to be set to the –fps. |
| **InterlacedVideo** | -interlacedVideo | [0 - 1] | 0 | 1 : encoder will signal interlaced signal in the stream, 0 : assumes progressive signal |
| **SeperateFields** | -separate-fields | [0 - 1] | 0 | 1 : Interlaced input, application will separate top and bottom fields and encode it as progressive 0 : Treat video as progressive video |
| **HierarchicalLevels** | -hierarchical-levels | [0 – 5] | 3 | 0 : Flat  1: 2-Level Hierarchy  2: 3-Level Hierarchy  3: 4-Level Hierarchy  4: 5-Level Hierarchy  5: 6-Level Hierarchy  Minigop Size = (2^HierarchicalLevels) (e.g. 3 == > 7B pyramid, 2 == > 3B Pyramid) |
| **BaseLayerSwitchMode** | -base-layer-switch-mode | [0 - 1] | 0 | 0 : Use B-frames in the base layer pointing to the same past picture  1 : Use P-frames in the base layer |
| **IntraPeriod** | -intra-period | [-2 - 119] | -2 | Distance Between Intra Frame inserted. -1 denotes no intra update. -2 denotes auto. |
| **IntraRefreshType** | -irefresh-type | [1 – 2] | 1 | 1: CRA (Open GOP)  2: IDR (Closed GOP) |
| **QP** | -q | [0 - 51] | 25 | Quantization parameter used when RateControl is set to 0 |
| **LoopFilterDisable** | -dlf | [0 - 1] | 0 | When set to 1 disables the Deblocking Loop Filtering |
| **SAO** | -sao | [0 - 1] | 1 | When set to 0 the encoder will not use the Sample Adaptive Filter |
| **UseDefaultMeHme** | -use-default-me-hme | [0 - 1] | 1 | 0 : Overwrite Default ME HME parameters  1 : Use default ME HME parameters, dependent on width and height |
| **HME** | -hme | [0 - 1] | 1 | Enable HME, 0 = OFF, 1 = ON |
| **HMELevel0** | -hme-l0 | [0 - 1] | 1 | Enable HME Level 0 , 0 = OFF, 1 = ON |
| **HMELevel1** | -hme-l1 | [0 - 1] | Depends on input resolution | Enable HME Level 1 , 0 = OFF, 1 = ON |
| **HMELevel2** | -hme-l2 | [0 - 1] | Depends on input resolution | Enable HME Level 2 , 0 = OFF, 1 = ON |
| **SearchAreaWidth** | -search-w | [1 - 256] | Depends on input resolution | Search Area in Width |
| **SearchAreaHeight** | -search-h | [1 - 256] | Depends on input resolution | Search Area in Height |
| **NumberHmeSearchRegionInWidth** | -num-hme-w | [1 - 2] | Depends on input resolution | Search Regions in Width |
| **NumberHmeSearchRegionInHeight** | -num-hme-h | [1 - 2] | Depends on input resolution | Search Regions in Height |
| **HmeLevel0TotalSearchAreaWidth** | -hme-tot-l0-w | [1 - 256] | Depends on input resolution | Total HME Level 0 Search Area in Width |
| **HmeLevel0TotalSearchAreaHeight** | -hme-tot-l0-h | [1 - 256] | Depends on input resolution | Total HME Level 1 Search Area in Width |
| **HmeLevel0SearchAreaInWidth** | -hme-l0-w | [1 - 256] | Depends on input resolution | HME Level 0 Search Area in Width for each region, separated in spaces, the number of input search areas must equal to NumberHmeSearchRegionInWidth, and the sum must equal to HmeLevel0TotalSearchAreaWidth |
| **HmeLevel0SearchAreaInHeight** | -hme-l0-h | [1 - 256] | Depends on input resolution | HME Level 0 Search Area in Height for each region, separated in spaces, the number of input search areas must equal to NumberHmeSearchRegionInHeight, and the sum must equal to HmeLevel0TotalSearchAreaHeight |
| **HmeLevel1SearchAreaInWidth** | -hme-l1-w | [1 - 256] | Depends on input resolution | HME Level 1 Search Area in Width for each region, separated in spaces, the number of input search areas must equal to NumberHmeSearchRegionInWidth |
| **HmeLevel1SearchAreaInHeight** | -hme-l1-h | [1 - 256] | Depends on input resolution | HME Level 1 Search Area in Height for each region, separated in spaces, the number of input search areas must equal to NumberHmeSearchRegionInHeight |
| **HmeLevel2SearchAreaInWidth** | -hme-l2-w | [1 - 256] | Depends on input resolution | HME Level 2 Search Area in Width for each region, separated in spaces, the number of input search areas must equal to NumberHmeSearchRegionInWidth |
| **HmeLevel2SearchAreaInHeight** | -hme-l2-h | [1 - 256] | Depends on input resolution | HME Level 2 Search Area in Height for each region, separated in spaces, the number of input search areas must equal to NumberHmeSearchRegionInHeight |
| **ConstrainedIntra** | -constrd-intra | [0 - 1] | 0 | Allow the use of Constrained Intra, when enabled, this features yields to sending two PPSs in the HEVC Elementary streams |
| **RateControlMode** | -rc | [0 - 1] | 0 | 0 : CQP , 1 : VBR |
| **TargetBitRate** | -tbr | Any Number | 7000000 | Target bitrate in bits / second. Only used when Rate Control is set to 1 |
| **MaxQpAllowed** | -max-qp | [0 - 51] | 48 | Maximum QP value allowed for rate control use. Only used when Rate Control is set to 1. Has to be > = MinQpAllowed |
| **MinQpAllowed** | -min-qp | [0 - 50] | 10 | Minimum QP value allowed for rate control use. Only used when Rate Control is set to 1. Has to be < MaxQpAllowed |
| **LookAheadDistance** | -lad | [0 - 120] | 17 | When Rate Control is set to 1 it's best to set this parameter to be equal to the Intra period value (such is the default set by the encoder) |
| **SceneChangeDetection** | -scd | [0 - 1] | 1 | Enables or disables the scene change detection algorithm |
| **ImproveSharpness** | -sharp | [0 - 1] | 1 | This is a visual quality knob that allows the use of adaptive quantization within the picture and enables visual quality algorithms that improve the sharpness of the background. This feature is only available for 4k resolutions |
| **VideoUsabilityInfo** | -vid-info | [0 - 1] | 0 | enables or disables sending a vui structure in the HEVC Elementary bitstream |
| **HighDynamicRangeInput** | -hdr | [0 - 1] | 0 | When set to 1, signals HDR10 input in the output HEVC elementary bitstream and forces VideoUsabilityInfo to 1. |
| **AccessUnitDelimiter** | -ua-delm | [0 - 1] | 0 | SEI message, 0 = OFF, 1 = ON |
| **BufferingPeriod** | -pbuff | [0 - 1] | 0 | SEI message, 0 = OFF, 1 = ON |
| **PictureTiming** | -tpic | [0 - 1] | 0 | SEI message, 0 = OFF, 1 = ON. If 1, VideoUsabilityInfo should be also set to 1. |
| **RegisteredUserData** | -reg-user-data | [0 - 1] | 0 | SEI message, 0 = OFF, 1 = ON |
| **UnregisteredUserData** | -unreg-user-data | [0 - 1] | 0 | SEI message, 0 = OFF, 1 = ON |
| **RecoveryPoint** | -recovery-point | [0 - 1] | 0 | SEI message, 0 = OFF, 1 = ON |
| **TemporalId** | -temporal-id | [0 - 1] | 1 | 0 = OFF, 1 = Insert temporal ID in NAL units |
| **AsmType** | -asm | [-1 - 1] | -1 | Assembly instruction set (-1: Automatically select highest assembly instruction set supported, 0: non-AVX2, 1: AVX2) |
| **UseRoundRobinThreadAssignment** | -rr | [0 - 1] | 0 | For Dual socket systems running a Windows\* OS on systems with > 32 physical processors. When enabled, allows the encoder to run on both sockets |

# Best Known Configurations (BKC)

This section outlines the best known hardware and software configurations that would allow the SVT-HEVC encoder to run with the highest computational performance. For the CQP mode, the output bit stream will not change if these BKCs have not been applied.

## Hardware BKC:

The SVT-HEVC encoder is optimized for use on Xeon Scalable Platforms. For best multichannel encode, servers should be set up with at least one 2666 Mhz DDR4 RAM DIMM per RAM channel per socket. For example, a dual Xeon Platinum 8180 server is best set up with 12 x 2666 Mhz DDR4 RAM DIMM.

## Software BKC:

The SVT-HEVC encoder drives the best performance when restricting each channel to only one socket on either Windows\* or Linux\* operating systems. For example, when running four channels on a dual socket system, it’s best to pin two channels to each socket and not split every channel on both sockets.

### 10 bit Input YUV:

Due to the large size of 10-bit video, using the unpacked YUV format as shown in section 3.1 allows for the best performance of the encoder.

### Windows\* OS (Tested on Windows\* Server 2016)

Visual Studio 2015 offers Profile Guided Optimization (PGO) to improve compiler optimization for the application. The tool uses an instrumented build to generate a set of profile information of the most frequently used code and optimal paths. The profile is then used to provide extra information for the compiler to optimize the application. To take advantage of PGO, build using the following:

1. Open the solution file with Visual Studio 2015 and build code in Release mode
2. Right click ebHevcEncApp project from the Solution Explorer -> Profile Guided Optimization -> Instrument (Repeat for ebHevcEncLib)
3. Right click ebHevcEncApp project from the Solution Explorer -> Properties -> Debugging
4. Add configuration parameters and run encoder (e.g. 1280x720 video encode of 300 frames)
5. Right click ebHevcEncApp project from the Solution Explorer -> Profile Guided Optimization -> Run Instrumented/Optimized Application
6. Right click ebHevcEncApp project from the Solution Explorer -> Profile Guided Optimization -> Optimize (Repeat for ebHevcEncLib)

### Linux\* OS (Tested on Ubuntu\* Server 16.04)

Some Linux\* Operating systems and kernels assign CPU utilization limits to applications running on servers. Therefore, to allow the application to utilize up to ~100% of the CPUs assigned to it, the following commands are best to be ran before and when running the encoder:

* + sudo sysctl -w kernel.sched\_rt\_runtime\_us=990000
    - this command should be executed every time the server is rebooted
  + Run the encoder application with sudo privileges which will allow the encoder to create threads with a real-time priority.

The above section is not needed for Windows\* as it does not perform the CPU utilization limitation on the application.

# Legal Disclaimer

Intel, Xeon are trademarks of Intel Corporation in the U.S. and/or other countries.

Benchmark results were obtained prior to implementation of recent software patches and firmware updates intended to address exploits referred to as "Spectre" and "Meltdown". Implementation of these updates may make these results inapplicable to your device or system.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to www.intel.com/benchmarks. Intel technologies’ features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.