CONVINcE
D2.2.2

Specification of power efficient encoder-transcoder

Editor: **Raoul Monnier (TVN)**

Reviewers: **Mikko Uitto (VTT)**

Authors: **Raoul Monnier (TVN),**

Khaled Jerbi (TVN)

Mikko Uitto (VTT)

EXECUTIVE SUMMARY

It has been identified in document D2.1.1 that the encoders/transcoders are the most energy and power consuming parts of a headend system. This document focuses on this part of a video headend by studying the architecture elements of video coder/transcoder and their impact on power consumption. It then proposes specifications of encoder/transcoder architecture with a focus on power efficiency. Alternative implementations are studied based on different technologies.

To achieve this we first identified the parameters affecting the power consumption in video encoding/transcoding. Power consumption parameters can be split in two categories:

- Non-encoding related: platforms and architectures.
- Encoding related: codecs and their implementation.

In the platform area processor manufacturers have developed a number of technology blocks that helps reduce consumption: parallelization and optimization of computing cores, clock and frequency scaling and hyper-threading.

GPU (Graphical Processing unit) hardware assisted decoding and encoding are another power optimization techniques that is available inside (embedded) or outside modern processors (stand-alone GPU).

The complexity of the source video (spatial and/or temporal) also affects the consumption.

Non encoding related parameters are analyzed in this document, where comparison between two encoding machines is done.

In the area of encoding related parameters we studied codecs internals and their relationship with consumption. Energy consumption can be impacted by compression parameters, output bitrate, parallelization, images processing and codecs implementations.

An analysis of theses parameters is then done with a focus on industry standards such as H264-AVC (Advanced Video Coding) and H265-HEVC (High Efficiency Video Coding) codecs and their different implementations. Objective video measurements and consumption measurements are performed with different codecs (x264 and x265). The results show opposite curves between energy consumption and output bitrate and/or image quality.

Parallelization also reduces the energy consumption because it reduces the encoding time for a fixed bitrate.

Video image processing (known as pre-processing) shows no impact on consumption as it may be always part of the image processing chain in the tested implementations.

Different codecs implementations were tested with a constant parameters set. X264, x265 and Kvazaar codecs are compared. Kvazaar consumes 13% more than x265 and they are both 4 times more consuming than x264.

Conclusion: HEVC consumes 4 times more energy than AVC and selection of parameters is a general trade-off between energy, output bitrate and image quality.

Encoders/transcoders architecture is also studied in this document where the consumption model of a transcoding appliance is defined. The model is represented by a fixed part (idle consumption) and a variable part increasing linearly for each video channel.

In transcoder architecture video encoder and video decoder are identified as the most consuming functional blocks.

A solution is proposed to reduce the energy consumption of these blocks by using GPU accelerated decoding and encoding. GPU decoding and encoding increase density per appliance and thus reduce energy consumption per video channel. An innovative approach called "Just In Time Transcoding" is then presented. It should give promising energy consumption savings as it reduces drastically the amount of video to be stored in a CDN (Content Delivery Network).

Image filtering before encoding (known as pre-processing) is also a functional block where we can leverage on consumption. Different technologies can be used to implement these algorithms from full hardware to full software.

The actions/gains can be summarized in the following table:

Actions	**Gains**
Tailor the platform	Static power reduction and per channel power reduction
Optimize the codec choice and parameters	Power reduction by video
Reduce the output bitrate	Network power consumption reduction
Optimize transcoder architecture	Per channel power reduction
Migrate transcoders to the edge	Reduction of video amount stored in CDNs

Table of Contents

Table of Figures

Table of Tables

1 DOCUMENT HISTORY AND ABBREVIATIONS

1.1 Document history

Version	Date	Description of the modifications
0.1	09/02/17	Draft of ToC (TVN)
0.2	23/02/17	Contribution from TVN
0.3	23/02/17	Contribution from VTT
0.4	27/02/17	Integrated version
0.5	07/03/17	Reviewed version
1.0	08/03/17	Final version

1.2 Abbreviations

2D	Two-Dimensional
3D	Three Dimensional
ABR	Adaptive Bit Rate
AMD	Advanced Micro Devices, Inc.
API	Application Programming interface
ARM	Advanced RISC (Reduced Instruction Set Computer) Machine
ASIC	Application-Specific Integrated Circuit
AVC	Advanced Video Coding (H264)
CBR	Constant Bit Rate
CDN	Content Delivery Network
CPU	Central Processing Unit
DASH	Dynamic Adaptive Streaming Over HTTP
DPB	Decoded Picture Buffer
DVFS	Dynamic Voltage and Frequency Scaling
ePDU	Enclosure Power Distribution Unit (Powerware; Eaton)
EU	Graphical processor Execution Units
FPGA	Field-Programmable Gate Array
GOP	Group Of Pictures
GPU	Graphical Processing unit
HD	High Definition
HDD	Hard Disk Drive
HEVC	High Efficiency Video Coding (H265)
HLS	HTTP Live Streaming (Apple)
HTTP	Hypertext Transfer Protocol
IDR	Instantaneous Decoding Refresh
IPTV	IP mode television content delivery (managed network).
MPEG	Moving Picture Experts Group (International Organization for Standardization/International Electrotechnical Commission)
Open CL	Open Computing Language
OpenCL	Open Computing Language
OpenGL	Open Graphics Library
OTT	Over The Top content delivery (unmanaged network).
PC	Personal Computer
PPS	Picture Parameter Set
PSNR	Pondered Signal to Noise Ratio
QoE	Quality of Experience
RTP	Real-Time Transport Protocol

SD	Standard Definition
SPS	Sequence Parameter Set
SS	Smooth Streaming
UDP	User Datagram Protocol
VBV	Video Buffer Verifier
VC1	Video Codec 1 (Windows Media 9 compression)
VHDL	VHSIC (Very High Speed Integrated Circuit) Hardware Description Language
VHDL	Very High-level Design Language
VP8, VP9	Open and royalty free video compression formats owned by Google
webm	Video file format (primarily intended to offer a royalty-free alternative to use in the HTML5 video tag)
YUV	Luminance-Bandwidth-Chrominance

2 INTRODUCTION

In the global goal to reduce the power consumption of video head-ends we already identified the transcoders as potentially the most consuming part. We now focus on encoders/transcoders regarding their energy impact.
In the first version of the deliverable, we categorized and identified the parameters which impact the power consumption. In this updated version of the document we are going further and we identify opportunities for saving power using novel processing architectures, new video coding standards such as HEVC and VP9 and also system architecture. We also illustrate the effect of power consumption alignment with content encoding difficulty.

3 POWER CONSUMPTION PARAMETERS

This chapter deals with all the parameters that affect the energy consumption in an encoder – transcoder.

3.1 Non encoding related parameters

These parameters affect the energy consumption but are not directly related to encoding schemes.

3.1.1 Platform

3.1.1.1 *Processor related parameters*

There exist several processor related parameters that may or may not have an effect on the encoding power consumption. These include:

- Processor architecture
- Number of used processor cores
- Processor clock frequency and frequency scaling
- Hyper threading
- Turbo boost

Different processor architectures, such as the ones used by Intel, ARM or AMD are worth inspecting. By changing the number of used processor cores it is possible to inspect power reduction possibilities. For example, with a lower number of used processor cores the power consumption can be reduced with the expense of increased encoding time.

Also, with a dynamic change of processor clock frequency less power may be consumed but again the encoding time is increased with lower clock frequency. This concept is generally defined as Dynamic Voltage and Frequency Scaling (DVFS). For example, Intel has implemented a technology called SpeedStep while AMD has a similar PowerNow technology. DVFS technologies are traditionally meant for laptop computers to save power.

Hyper threading is a simultaneous multithreading implementation developed by Intel. Its purpose is to improve the execution of multiple tasks at once performed on Intel's x86 microprocessors. In hyper threading, two processing threads per physical core are delivered. Multithreaded tasks benefit the most out of hyper threading as they get more work done in parallel. Hyper threading allows the usage of smaller number of processor cores and increases also the performance of single threaded tasks.

Turbo boost is a technology developed by Intel to dynamically increase the processor clock frequency when more computational power is needed. The concept is generally considered as dynamic overclocking, in which the processor is enabled to run above its base clock frequency. Once the need for extra computational power is over, the clock frequency is returned to the baseline level.

3.1.1.2 *GPU related parameters*

GPU decoding:

GPU decoding is now a "wired" function in the embedded GPU part (called *HD Graphics* or *Iris Pro graphics*) of Intel processors. Decoding functions exhibit very low power requirements allowing a large number of decoding in one processor. The only limitation is the bandwidth to input many streams in parallel.

Table 1: GPU hardware decoding by Intel architecture

architecture	Codec decoded in hardware
Haswell	MPEG2, H264/AVC, VC1
Broadwell	MPEG2, H264/AVC, VC1, VP8, partial VP9
Skylake	MPEG2, H264/AVC, VC1, VP8, VP9, H265/HEVC

GPU encoding:

GPU encoding is also a hardware accelerated function in the embedded GPU of Intel processors. Encoding is not as advanced in Intel architecture as decoding.

Table 2: GPU hardware coding in hardware

architecture	Codec encoded in hardware
Haswell	MPEG2, H264/AVC (partial)
Broadwell	MPEG2, H264/AVC
Skylake	MPEG2, VP8, H264/AVC, H265/HEVC, VP9 (partial)

Encoding may be done using 2 methods:

- Fully wired encoder which does the whole encoding loop with dedicated logic.
- Encoding done by using EU (Execution units) of the GPU very similar to OpenCL implementation. EU are general purpose parallel processing units which can be used either for 3D rendering or other video processing (in OpenGL).

As these graphical units are tailored to handle video processing the power requirements are quite lower than what is needed for CPU encoding.

3.1.2 Video type

Video type can also affect the power consumption of encoding. For example, encoding a video with a lot of motion and action, such as an action movie or a sports game, may consume more power than encoding a video with limited movement.

3.2 Encoding related parameters

Both H.264-AVC and H.265-HEVC standards hold several encoding tools that affect the energy consumption of the encoding process. In this deliverable, we focus on the following parameters: encoding presets, bit rate, number of threads and deblocking filter. Moreover, different encoder implementations have an effect on the power consumption.

3.2.1 Encoding presets

An encoding preset is defined as a set of parameters and their values. Presets are closely related to a specific encoder implementation. For example, both x264 and x265 encoders have specific encoding presets implemented. The purpose of the presets is to make a trade-off between the encoding speed and the compression efficiency.

In general both x264 and x265 encoders hold ten presets. The lowest preset is "veryslow" and the highest is "ultrafast". With "veryslow", the encoder tries to achieve the best quality per bit compression ratio with the expense of increased encoding time. "Ultrafast" preset does the encoding process as fast as possible, but with the expense of quality and compression efficiency.

The specifics of the presets and the parameter values can be found from the x265 documentation [1] and the Annex of this document. "Placebo" preset enables transform-skip prediction evaluation.

3.2.2 Bit rate

Bit rate determines the compression efficiency of the encoding process. With higher bit rate, the video file size is larger and with low bit rate the file size is smaller. Lower bit rates call for longer encoding time and higher bit rates for shorter encoding time.

3.2.3 Number of threads

The number of threads option allows creating threads to encode in parallel on multiple CPUs. With a multi-processor machine, using a higher number of threads can increase encoding speed linearly with the number of CPU cores. The default number of threads is set as 12.

3.2.4 Deblocking filter

A deblocking filter, i.e., a video filter is applied to decoded compressed video. The filter aims to improve visual quality, prediction performance and appearance of decoded pictures. Deblocking smooths sharp edges that can be formed between macroblocks when block coding techniques are used. Disabling the deblocking filter in video encoding may give reduced energy consumption.

3.2.5 Codec implementations

There exist several different encoder implementations for both H.264-AVC and H.265-HEVC. In this deliverable the most notable ones are considered. These include: x264 [2], x265 [3] and Kvazaar [4].

4 ANALYSIS OF THE POWER CONSUMPTION PARAMETERS

4.1 Test environment and scenario

The effect of the power consumption parameters has been analyzed with a test bed that is depicted in Table 3.

Table 3 : Test bed configuration

Parameter	Value
Server type	Dell PowerEdge R150 rack server
CPU	2 CPUs: Intel Xeon E5606, 4C, 2.13GHz
Memory	16 GB
HDD	1 TB
Operating system	Ubuntu Server
Power meter	Eaton Managed ePDU device

A single rack server is used for encoding video files and an Eaton Managed ePDU device is used for measuring power.

The test scenario involves encoding a single raw video file (YUV format) into an mp4 container. The parameters of the video file can be seen in Table 4.

Table 4 : Video file parameters

Parameter	Value
Length	43s 867ms
Frame rate	30 fps
Width	1920 pixels
Height	1080 pixels
Color space	YUV
Chroma subsampling	4:2:0

4.2 Non encoding related parameters

In addition to encoding related parameters, we wanted to see the effect when using more powerful processing architecture in the full software video encoding. Furthermore, content complexity (spatial/temporal) affects to power consumption. Therefore three different source inputs were used: low, hard and mixed (animated end credits) encoding difficulties, respectively.

Table 3 and Table 5 show the used test beds in our comparison. Both contain identical encoding software. Table 6 presents the input video sequences, which were extracted and selected from Tears of Steel open-source video clip. x264 and x265 software encoders were used for encoding the clips to H.264 and HEVC formats.

Table 5 : Test bed 2 configuration

Parameter	Value
Server type	Intel Core i7-5820K
CPU	6 CPU cores @ 3.3 GHz
Memory	32 GB
Operating system	Ubuntu 16.04
Power meter	Eaton Managed ePDU device

Table 6 : Video sequence parameters

Parameter	Sequence 1	Sequence 2	Sequence 3
Coding difficulty	Low	Hard	Mixed
Sequence name	Tears of Steel	Tears of Steel	Tears of Steel
Length (s)	60	60	60
Frame rate (fps)	24	24	24

Resolution		1920x1080	1920x1080	1920x1080
Color space		Y4M	Y4M	Y4M
Bitrate (Mbps)	**H.264/HEVC**	5,3 / 1,8	9,1 / 3,0	13,5 / 5,5
PSNR-Y (dB)		42,9	41,1	37,3

Figure 1 and Figure 2 show the encoding comparison results for the both machines. As can be seen, the more powerful Core i7 PC consumes less power than its Xeon reference. For H.264, the power saving is approximately 30% and for HEVC 35%.

The results also show that video source complexity affects to power consumption. As the sequence 2 contains lots of motion, it also requires more work from the compression algorithms and therefore increases also power consumption.

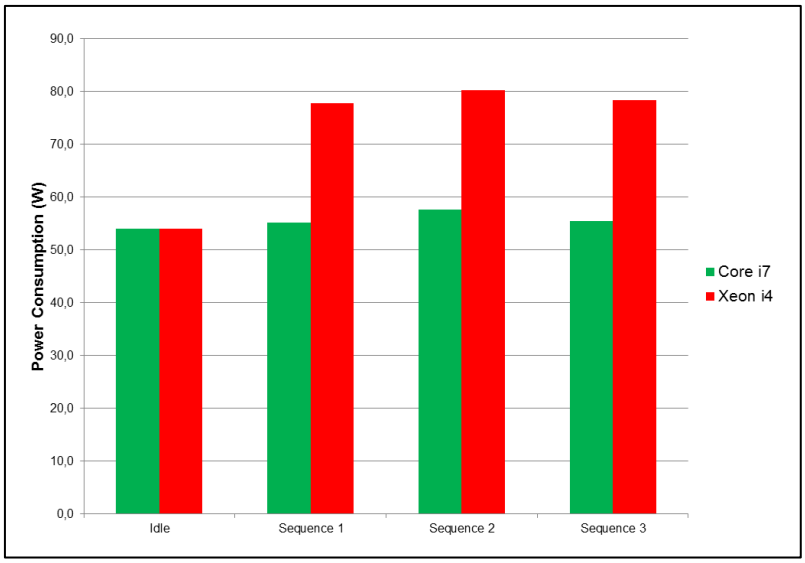

Figure 1: H.264 comparison results for test bed machines.

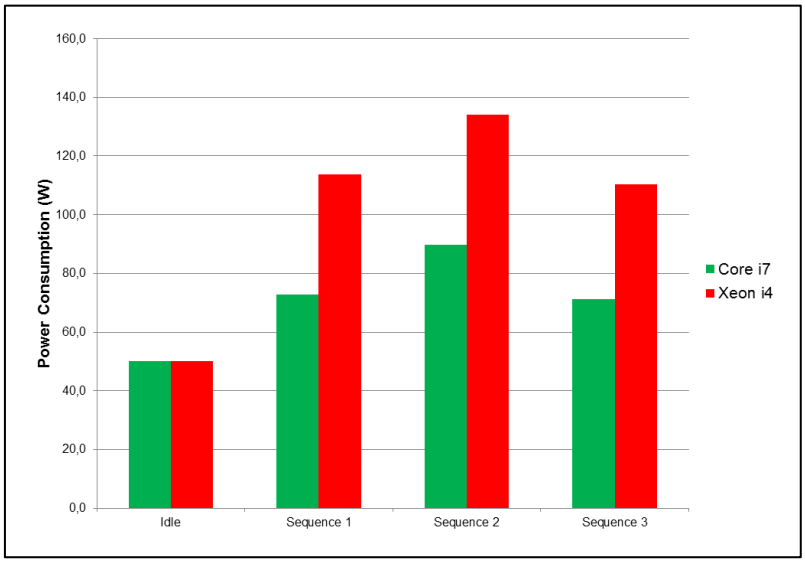

Figure 2: H.265 comparison results for test bed machines.

4.3 Encoding related parameters

4.3.1 H.264/AVC encoding

4.3.1.1 *Encoding presets*

Figure 3 shows the results with different encoding presets with the x264 encoder. In this case the PSNR values were roughly the same with no large deviations. The PSNR can thus be considered as constant and the energy and bit rate as variables. The results show that the energy consumption is smaller with the higher presets, such as ultrafast, as the encoding is done as fast as possible. However, the average bit rate is consequently larger with the higher presets. With the lower presets the energy consumption is larger but bit rate is smaller.

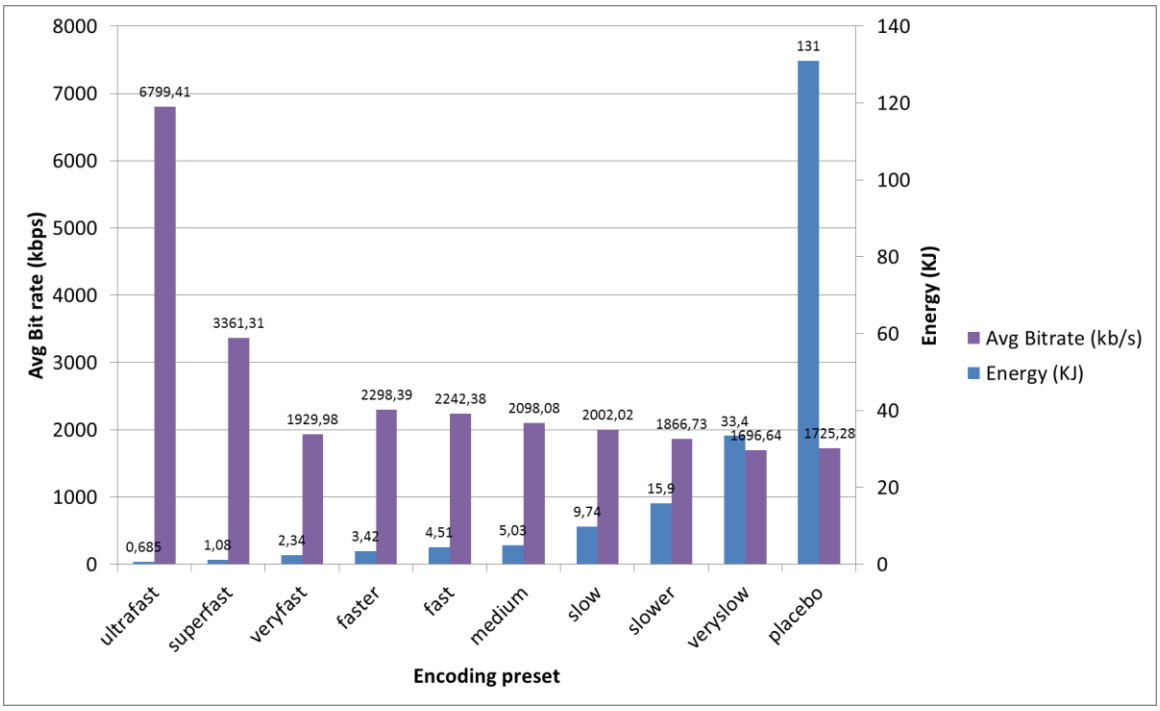

Figure 3 : H.264 results with different encoding presets.

4.3.1.2 *Bit rate*

Figure 4 shows the results when the bit rate is kept at a constant. In these experiments the bit rate was roughly 2 Mbps with very small variations. The results again show that with higher presets the energy consumption is smaller and with the smaller presets the energy consumption is larger. However, with the higher presets the average PSNR value is lower.

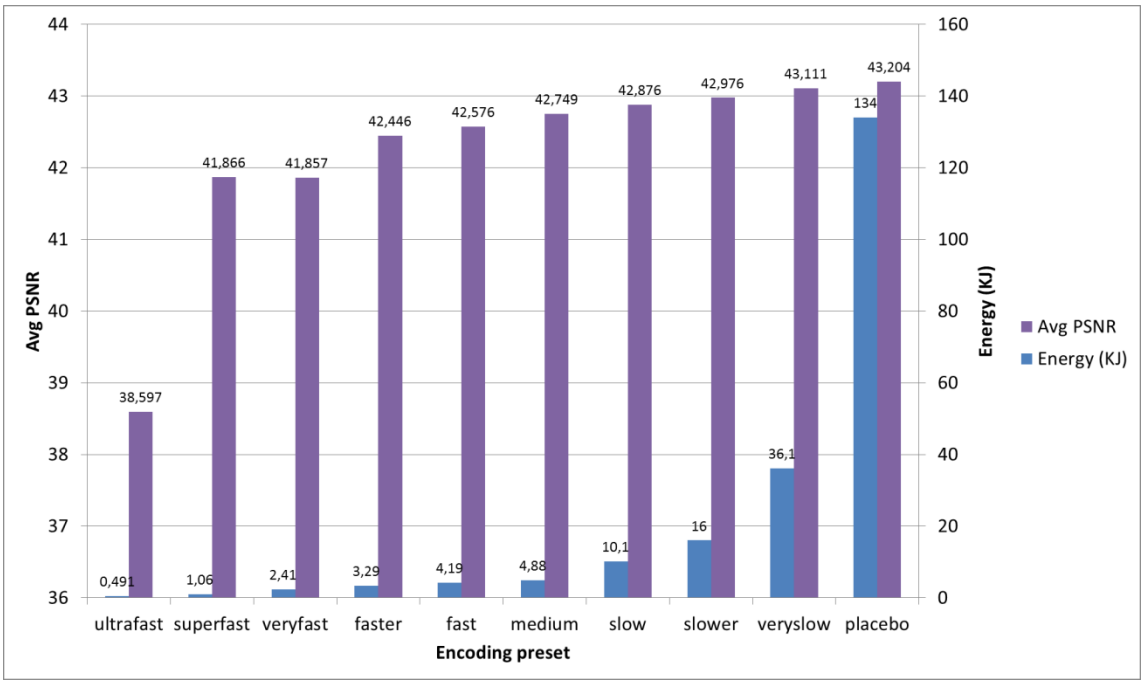

Figure 4 : H.264 results with constant bit rate.

4.3.1.3 *Number of threads*

Figure 5 shows the results with the number of threads. The experiments were done with the medium preset of the x264 encoder. The figure points out that the energy consumption decreases linearly as the number of threads increases. This is because increasing the number of threads decreases the encoding time by utilizing more processor cores and less energy is thus consumed. Using a single thread only a single processor core is used.

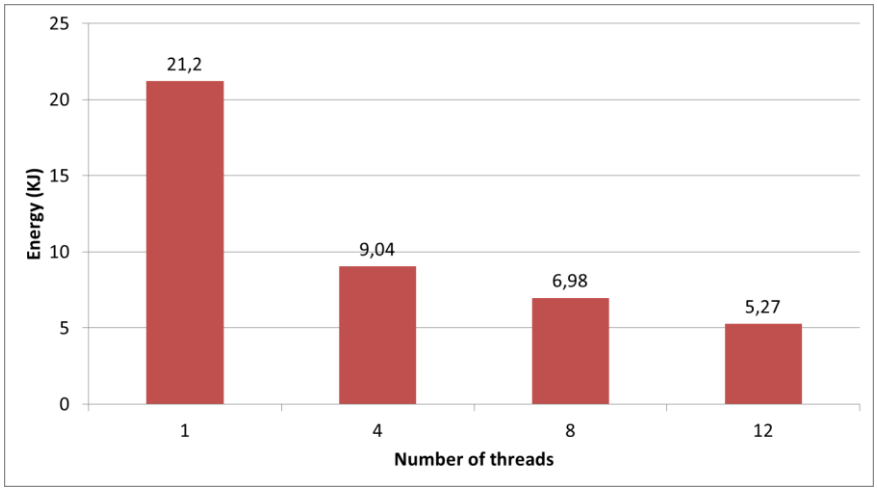

Figure 5 : H.264 results with number of threads.

4.3.1.4 *Deblocking filter*

The results with the deblocking filter can be seen in Figure 6. The results are an average of 10 experiments done with the medium preset of x264 encoder and they show that disabling the deblocking filter does not have an effect on the energy consumption. This is because the deblocking filter is an inherent part of the encoding-decoding loop.

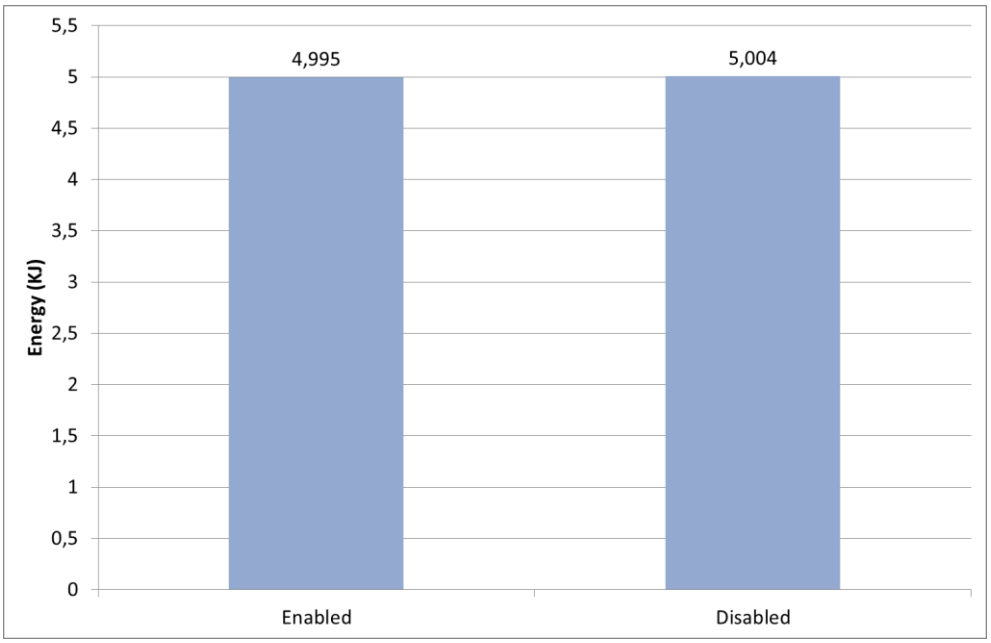

Figure 6 : H.264 results with the deblocking filter.

4.3.2 H.265/HEVC encoding

4.3.2.1 *Encoding presets*

Figure 7 shows the results with different encoding presets with the x265 encoder. The results are similar as with H.264: with higher encoding presets the energy consumption is smaller but the bit rate is larger.

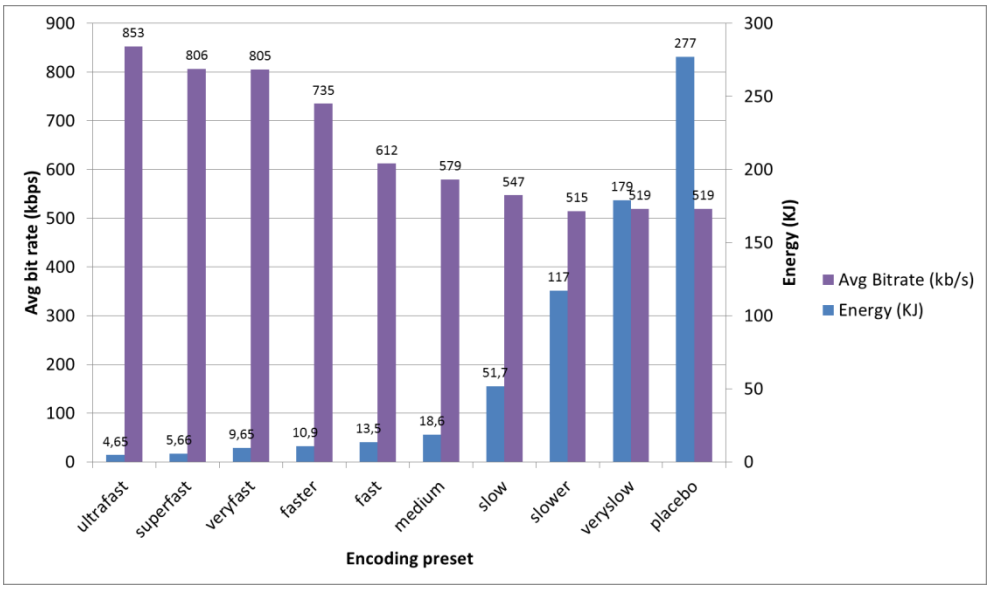

Figure 7 : H.265 results with different encoding presets.

4.3.2.2 *Bit rate*

The results with constant bit rate are shown in Figure 8. A bit rate of approximately 600 kbps was used. The energy consumption is smaller with higher presets and larger with smaller presets. However, the average PSNR value is lower with the higher presets.

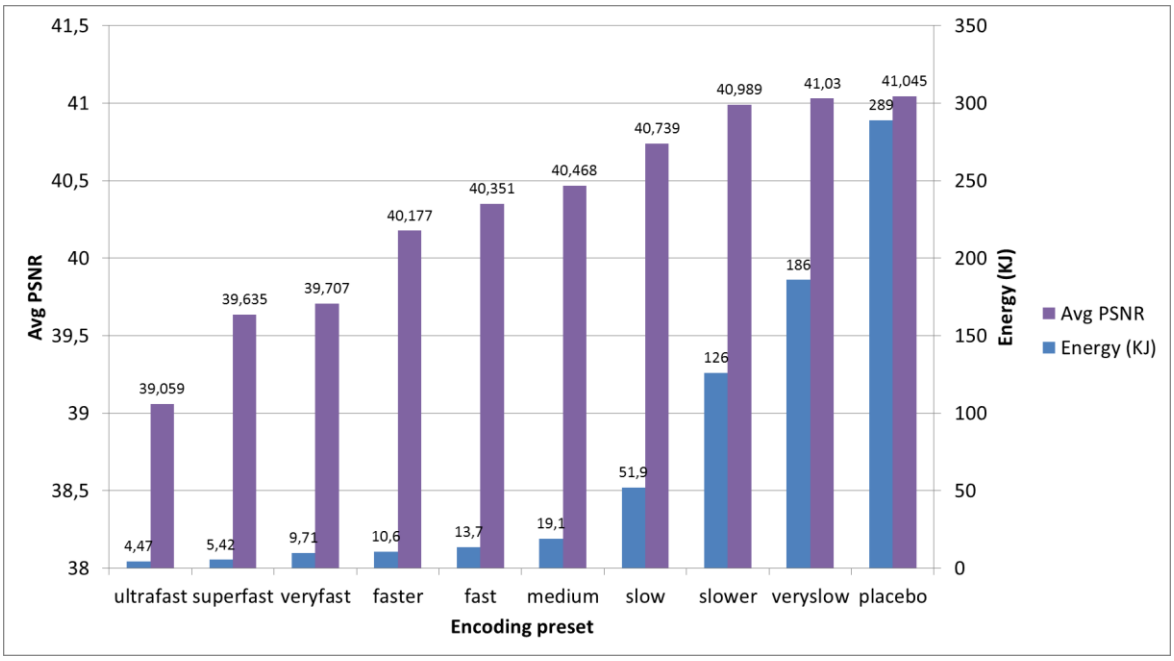

Figure 8 : H.265 results with constant bit rate.

4.3.2.3 *Number of threads*

Figure 9 illustrates the results with number of threads. The energy consumption decreases when number of threads increases as using a higher number of threads decreases the encoding time. The reduction in energy consumption is, however, not so drastic after using more than 4 threads. This is because H.265 encoder utilizes cores evenly regardless of number of threads. The encoding was done with the medium preset of the x265 encoder.

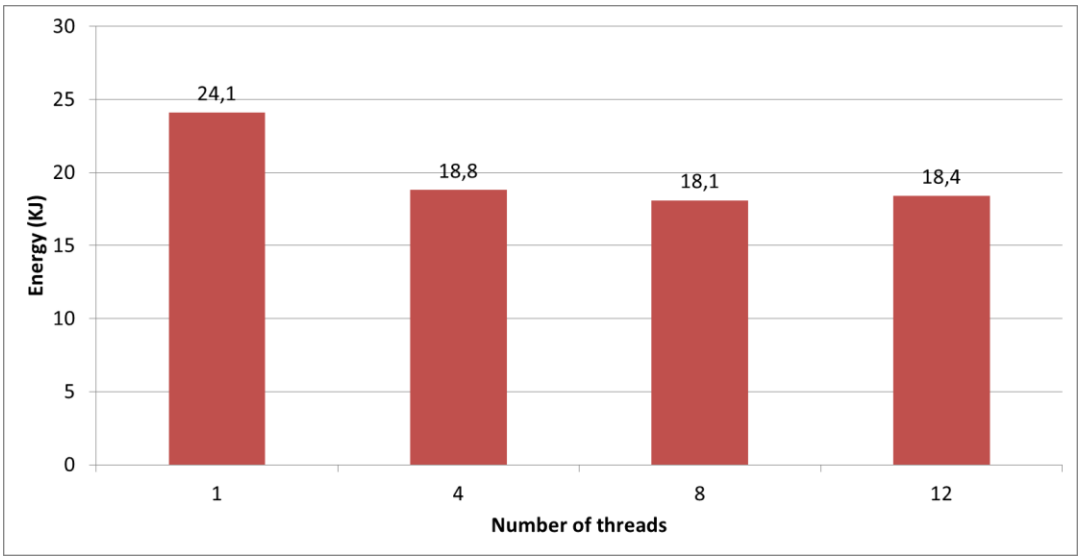

Figure 9 : H.265 results with number of threads.

4.3.2.4 *Deblocking filter*

Figure 10 shows the results with the deblocking filter. The results are an average of 10 experiments done with deblocking filter enabled and disabled. As with H.264, disabling the deblocking filter does not reduce the energy consumption as the deblocking filter is an inherent part of the encoding-deblocking loop. The encoding was done with the medium preset of the x265 encoder.

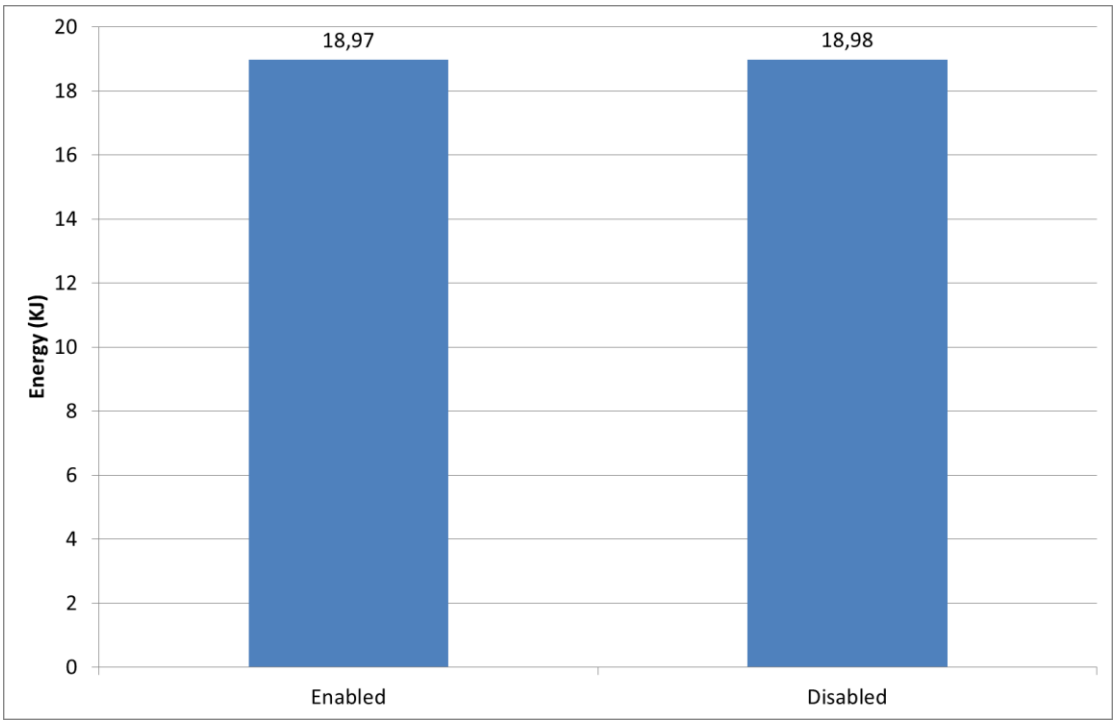

Figure 10 : H.265 results with deblocking filter.

4.3.3 Different codec implementations

The results with different encoders are shown in Figure 11. The experiments were done with x264, x265 and Kvazaar encoders and the results are an average of 10 experiments. The x265 encoder consumes on average 13 % less energy than the Kvazaar encoder. However, the Kvazaar encoder is still at an early stage of development and is improving gradually. When comparing HEVC with AVC, HEVC consumes approximately 4 times more energy than AVC in encoding.

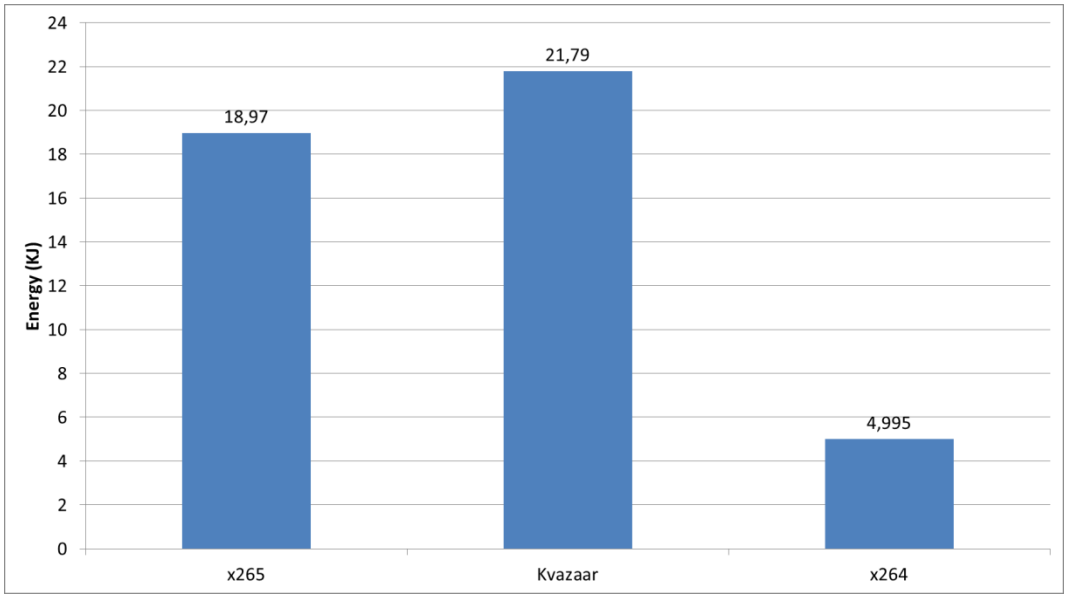

Figure 11 : Results with different encoder implementations.

The experimental results with different parameters show that selection of the parameters is in general a trade-off between energy, bit rate and PSNR. By selecting parameters that reduce the encoding time the energy consumption is consequently reduced. However, this comes with the expense of larger bit rate as the video compression is not optimal. Finally, if both encoding time and bit rates are tried to keep to a minimum, it comes with the expense of lower PSNR values.

Figure 12 show the power consumption comparison between H.264, H.265 and VP9 encoding formats. Two software encoders for HEVC (x265 and Kvazaar) were evaluated. As can be seen in terms of power consumption vs quality, VP9 performs surprisingly well. However, VP9 using webm format is still lacking terminal playback support in many devices especially when using MPEG-DASH streaming format. For more information about test bed configuration, please see [5].

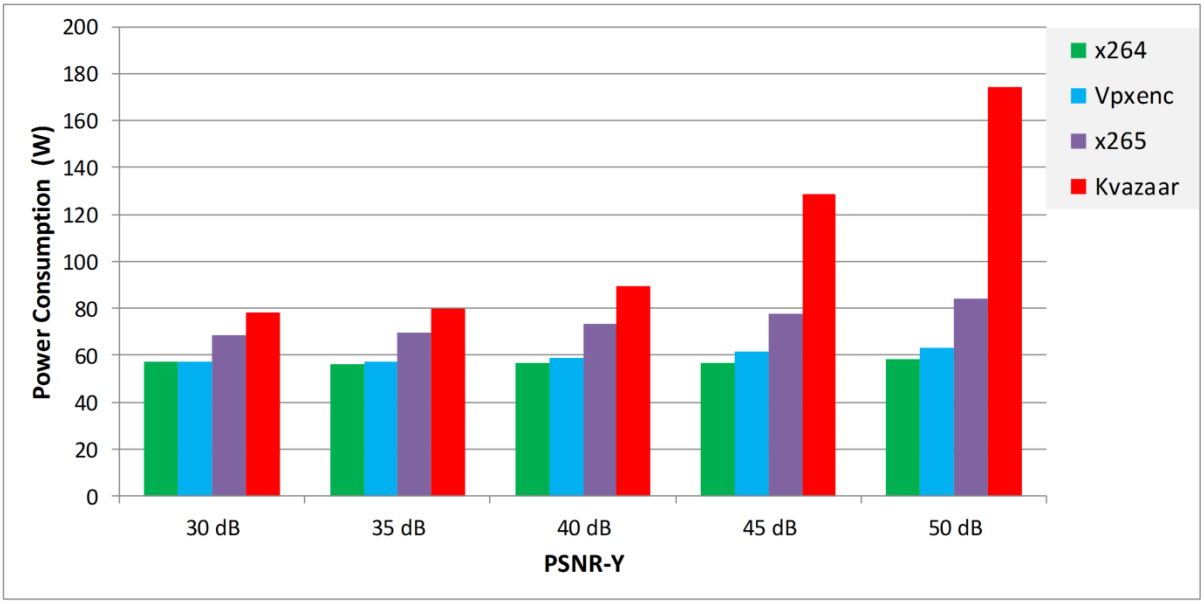

Figure 12: Power consumption comparison between x264, x265, Vpxenc and Kvazaar [5].

5 POWER EFFICIENT ENCODER-TRANSCODER

5.1 Architecture

A video channel represents the elements of a TV program. It is composed of:

- A single video stream.
- An arbitrary number of audio channels (starting from 1 stereo channel).
- Optional metadata streams such as subtitles, conditional access data and private data.

The consumption model of encoder-transcoder appliances is composed of:

- A fixed part (or idle consumption) with the following blocks:
  - The hardware platform (server based).
  - The operating system.
  - The management parts of the application software which is not related to the video chain.
- A variable part increasing linearly for each video channel.

The resulting power consumption model is as follow:

$$Ptrans_{(Wh)} = Pfixed_{(Wh)} + \left(Pchannel_{(Wh)} \times Number\ of\ channels\right)$$

5.1.1 Full software decoding/encoding

Full software implementation relies only on the CPU computational power to realize video processing tasks as well as decoding and encoding tasks.

The main advantages of full software solutions are flexibility and video quality.

Figure 13 shows the typical transcoding workflow for a channel in an IPTV application:

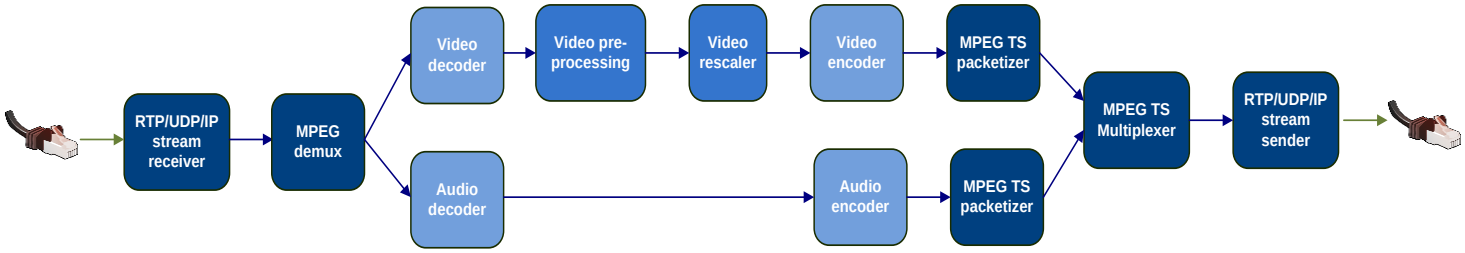

Figure 13: transcoding workflow full software - IPTV

Figure 14 shows the typical transcoding workflow for a channel in an OTT application:

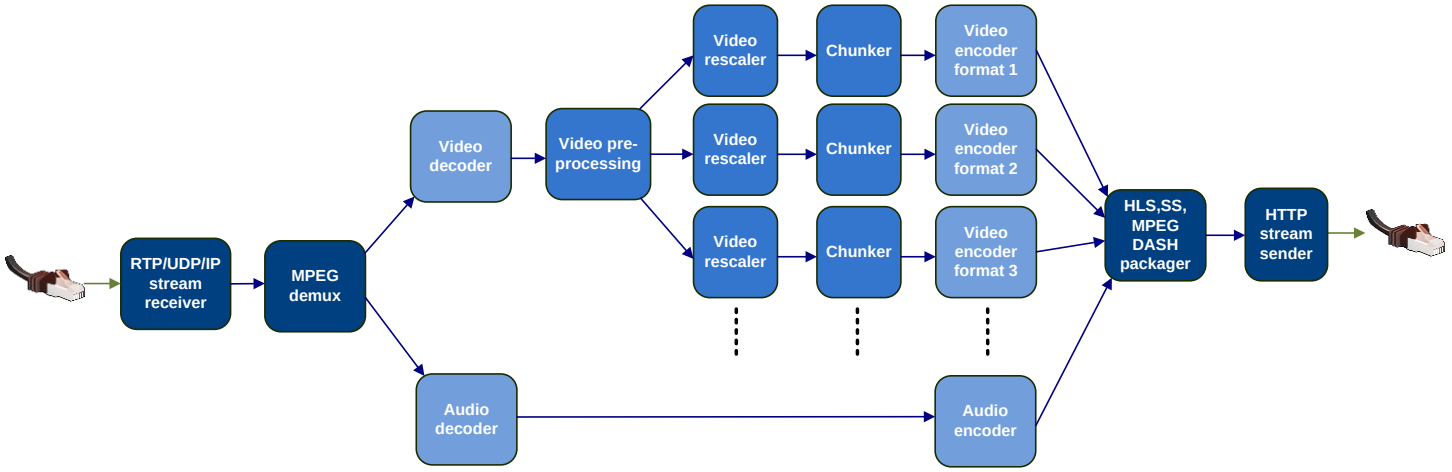

Figure 14: transcoding workflow - full software - OTT

The most consuming parts in these workflows are (by decreasing power):

- The video encoder
- The video decoder
- The audio encoder
- The audio decoder
- The video pre-processing, video rescaler
- The other functional blocks

The focus should then be put on video encoding and decoding functions.

5.1.2 GPU accelerated decoding/encoding

GPU accelerated functions rely on ASIC based functionalities related to video inside recent processors. The emphasis here will be put on Intel processors though these functionalities exist in other brands (namely AMD). Intel processors have proven to be the most energy efficient general purpose processors for server appliances.

In this case video encoding and decoding functions are performed by Intel QuickSync Video GPU functionality. It consists in hardware blocks, a GPU driver (i915) and software API (VAAPI).

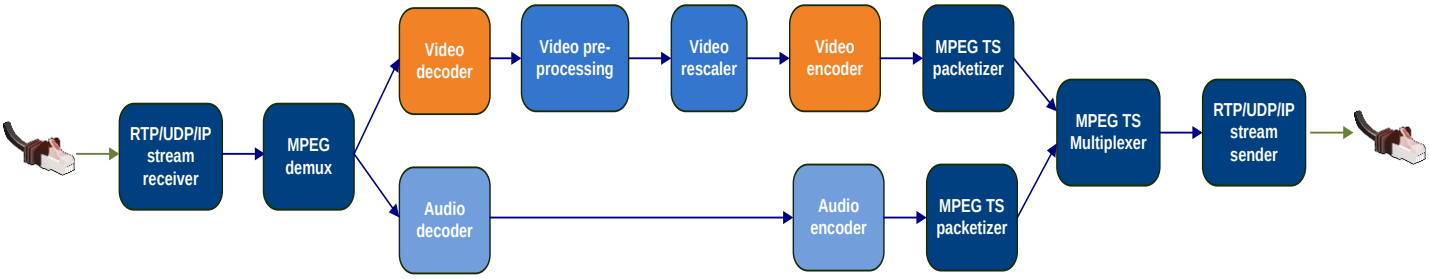

Figure 15: transcoding workflow GPU acc. – IPTV

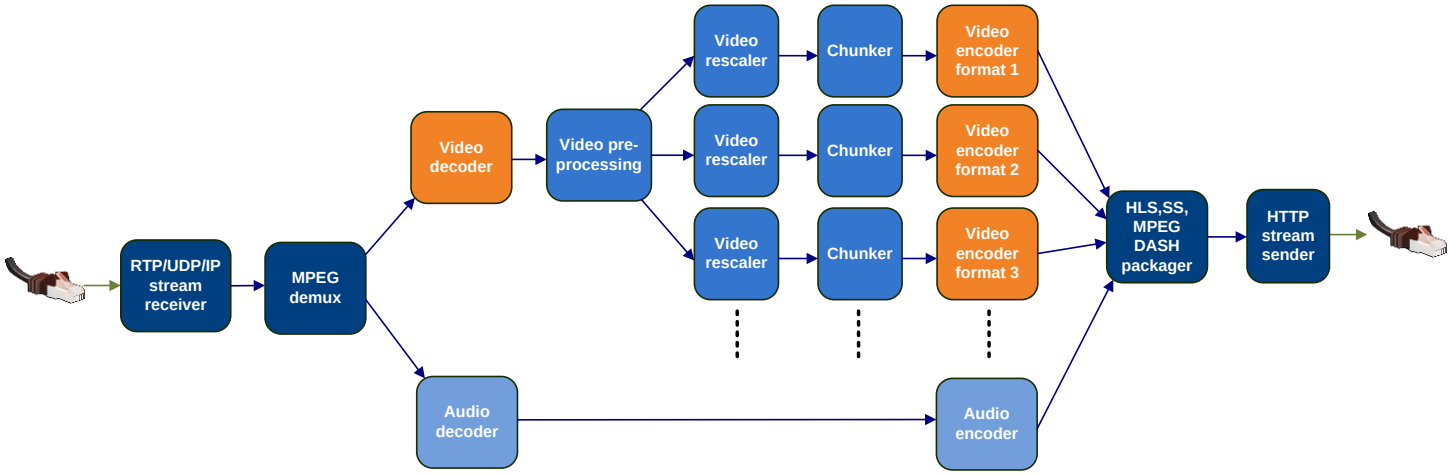

Figure 16: transcoding workflow - GPU acc. - OTT

In these applications the Video decoder and the video encoder use hardware functions from the GPU (QuickSync Video). See chapter 0 for capabilities.

The main advantage of GPU solutions is density (and thus consumption).

5.1.3 Just In Time Transcoding

"Just In Time Transcoding" is a concept where, instead of storing in a CDN a lot of representations of a single content in order to address different terminals and bit rates, video is stored in a single format (called "mezzanine" format) and transcoded at terminal request.

It is expected to save energy as we reduce significantly the energy cost for storing a huge number of representations.

Let's take Netflix example which can be considered as the state of the art for OTT delivery today. Netflix is storing in the CDN representations corresponding to the combination of:

- 10 resolutions to allow for bitrate adaptation to the terminal[1],
- 4 codecs, depending on the terminal the video is watched on[2] (VC1, H.264/AVC Baseline, H.264/AVC Main and HEVC),
- 3 ABR technologies (HLS, Smooth Streaming and DASH), in order, also, to address different terminals.

It leads to 120 representations[3] with the goal of streaming to more than 900 different devices. Figure 17 shows the complexity of such a traditional approach.

[1] http://techblog.netflix.com/2015/12/per-title-encode-optimization.html
[2] http://techblog.netflix.com/2015/12/high-quality-video-encoding-at-scale.html
[3] https://gigaom.com/2012/12/18/netflix-encoding/

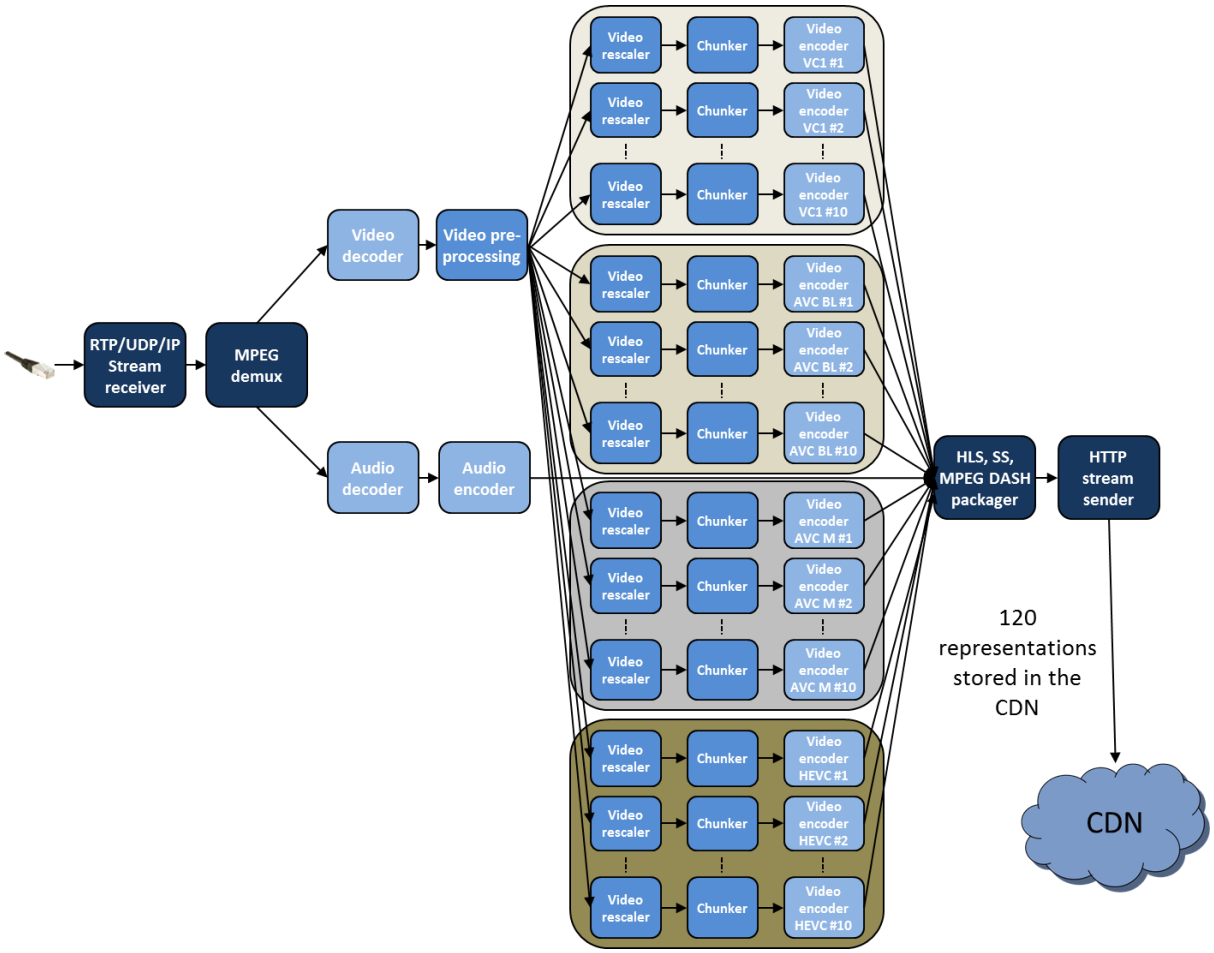

Figure 17: State of the art for OTT delivery – Netflix use case

"Just In time Transcoding" allows reducing drastically the number of representations to be stored in the CDN as depicted in Figure 18.

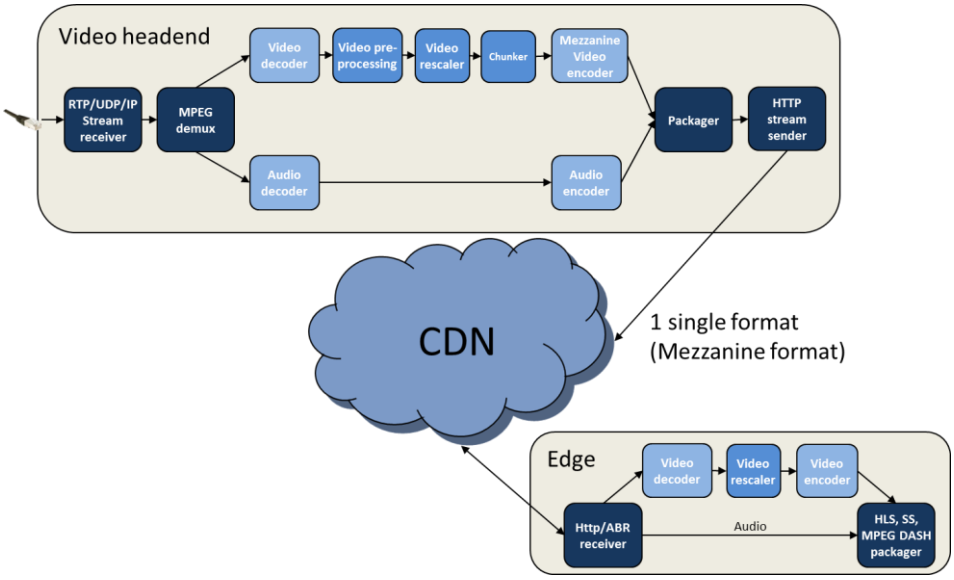

Figure 18: "Just In Time Transcoding" block diagram

In the headend, only one representation is produced in a single format (mezzanine format) it is stored in the cloud (CDN). In the edge, as close as possible to the end user, a transcoding

operation is processed in order to provide the terminal of the end user with the requested format/bitrate.

The variant shown by Figure 19 can also be used to reduce the power consumption, making use of metadata generated by the encoder in the headend and of a simplified video transcoder.

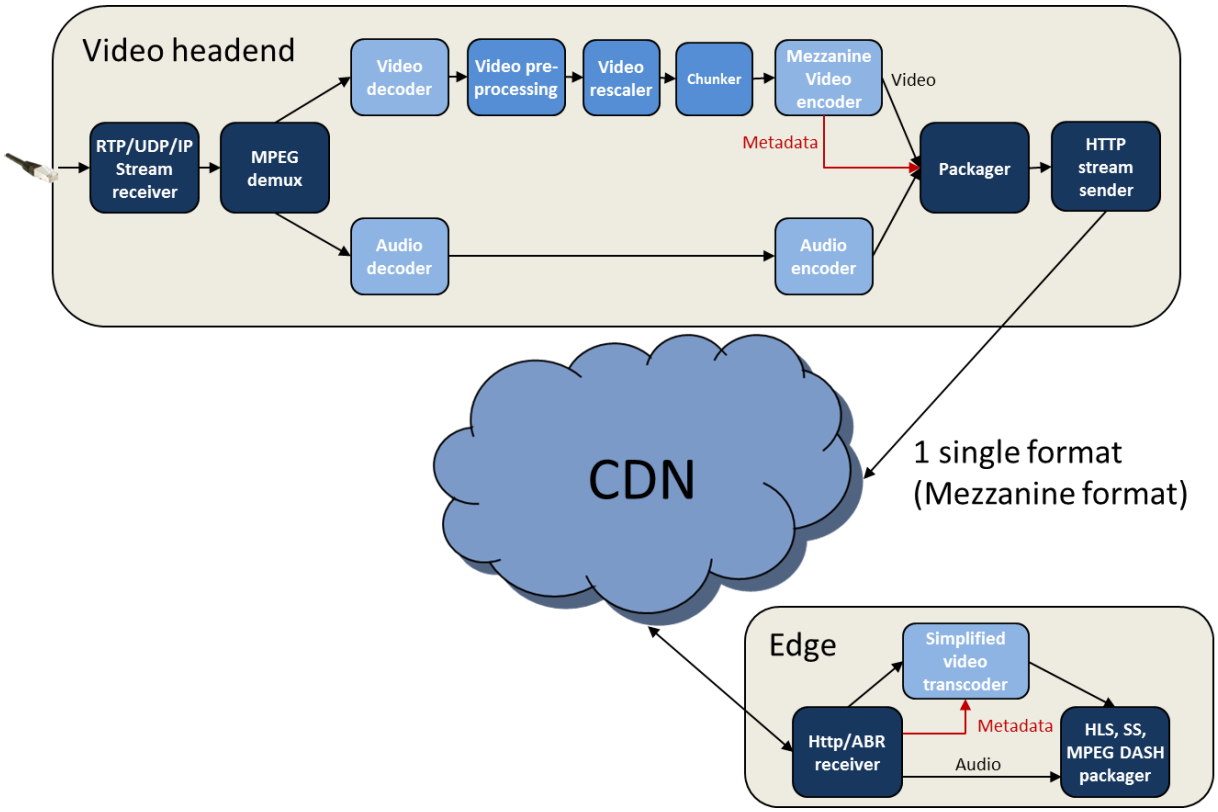

Figure 19: Optimized edge transcoding using metadata

This workflow is in line with work done by MPEG on "Network Distributed Video Coding" where draft requirements were issued during 115th MPEG meeting in Geneva[4]. The main focus of this Ad hoc Group is "guided transcoding" standardization. Figure 20 summarizes the approach suggested by MPEG, which is a generic view of the work done by Harmonic on "Just In Time Transcoding".

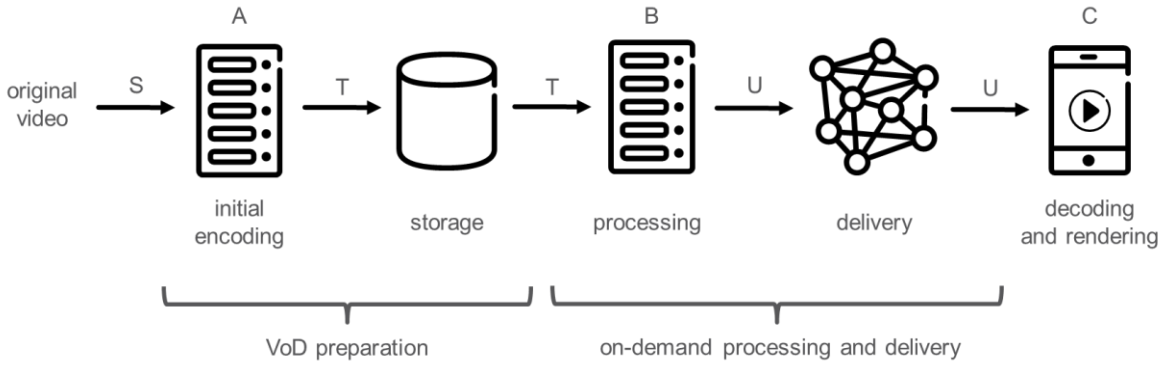

Figure 20: Guided transcoding in MPEG

[4] http://mpeg.chiariglione.org/standards/exploration/future-video-coding/draft-requirements-network-distributed-video-coding

5.1.4 Video pre-processing

Video pre-processing is the action of filtering individual image or a sequence of images in order to ease the encoding process or the decoding process (at the terminal).
Common processing functions are:

- Image rescaler/resizer
- Denoiser
- Deblocking filter
- 2D filter
- Mosquito noise reduction
- Color tone correction

Pre-processing algorithms can be implemented using different technologies (from hardware to software):

- ASIC blocks in GPUs.
- FPGA cells (VHDL).
- OpenCL in FPGA cells.
- OpenCL in GPUs.
- Full software.

5.2 Expected gains

By tailoring the encoder architecture we expect 2 advantages:

- Reducing the power consumption by channel.
- Increasing the density in number of channels by appliance.

For a platform with a bounded computing power (fixed number of processors/cores) these two advantages are linked.

By choosing the right codec we can optimize the consumption of the encoder for a constant output bitrate or a constant quality of experience.

NOTE: Reducing the bandwidth in the network may be counterproductive for transcoder consumption but could be a global consumption gain for the project (depending on the scale factor).

The "Just In Time Transcoding" approach will allow to reduce the number of video representations stored in the CDN and thus its power consumption. This is an end-to-end approach to be compared to the global abovementioned one consisting in reducing the bandwidth of the network by using a most efficient encoding technology (HEVC/AVC).

6 CONCLUSIONS

The parameters which impact the power consumption in video encoder/transcoder can be split in 2 categories: non-encoding related and encoding related.
In both categories we can leverage on parameters to reduce the power consumption.
We can then tailor:

- The platform,
- The choice of codecs (and their implementation),
- The transcoder architecture,
- The network architecture.

7 REFERENCES

[1] "x265 preset options," [Online]. Available: https://x265.readthedocs.org/en/default/presets.html.
[2] "x264 encoder," [Online]. Available: http://www.videolan.org/developers/x264.html.
[3] "x265 encoder," [Online]. Available: http://x265.org/index.html.
[4] "Kvazaar HEVC encoder," [Online]. Available: http://ultravideo.cs.tut.fi/#encoder.
[5] "Uitto, M.:" Energy consumption evaluation of H.264 and HEVC video encoders in high-resolution live streaming, in WiMob, 17-19 Oct. 2016

8 ANNEX

The Table 7 shows the detailed parameters of the x264 encoder. Recommended values are to be used for optimal encoding.

Table 7 : x264 parameters

X264 parameter	acronym	Explanation	Default value	recommended
FRAME				
keyint	Keyframe interval	GOP length/ IDR interval	250	10x framerate 2-5x framerate HD
Keyint_min	Keyframe interval min	Min length between IDR	25	
scenecut		Difference treshold betwen image to generate IDR	40	
Pre-scenecut		Enables a faster scenecut mechanism. It's enabled by default when you use more than one thread		
Bframes	B-frames max limit	maximum number of concurrent B-frames	16	
b-adapt		B strategy, number of B frames auto mode :0 :very-fast 1 : fast, default 2 : slow, accurate	1	1st pass only in multipass encode
b-bias		Controls the likelihood of B-frames being used instead of P-frames.	0	
b-pyramid	B-references insertion	Allow the use of B-frames as references for other frames. ~~Increase the decoding picture buffer~~		
No-cabac		Remove cabac from coding. →CAVLC *No cabac in baseline profile*	-	
Ref	Reference frames	Controls the size of the DPB (**D**ecoded **P**icture **B**uffer) ~~Number of reference frames used~~	1	720p max : 9 1080p max : 4 maximum ref = 12288 * 1024 / (width * height * 1.5) 4-6 recommended
No-deblock		Disable loop deblocking filter	off	
Deblock <alpha :beta>		In loop deblocking filter Alpha :strength Beta :threshold	0,0	
Interlaced		Enable interlaced encoding		
RATE CONTROL				
Qp (rate ctrl1)	**C**onstant **Q**uantizer mode P	**C**onstant **Q**uantizer mode (P frame quantizer). Target an certain quantizer. mutually exclusive with bitrate and crf.	Not used	
Bitrate (rate ctrl2)		Target bitrate mode. (CBR) Targets filesize. In 2 passmode only Something which gives P frame	bitrate in kilobits/sec .	

X264 parameter	acronym	Explanation	Default value	recommended
		quantizers around 18-26. Estimated ranges: for SD resolution: 800kbits - 2100kbits. For 720p, 3-6mbit. For 1080p, 8-15mbit+.		
Crf (rate ctrl3)	**C**onstant **R**ate**f**actor	**C**onstant **R**ate**f**actor. crf targets a certain 'quality' Best one pass option.		The range 18-26 is probably where you will want to look at. If you need absolutely perfect quality you could go down to 16
Vbv-maxrate		maximum bitrate in VBV mode. Requires VBV buvvfersize. 2 pass mode.		
Vbv-bufsize		Sets the maximum size of the VBV buffer. Only for harware device		
Vbv-init		Sets the initial size of the VBV filled buffer size.	0	
Qpmin	Minimum quantizer	Defines the minimum quantizer that x264 will ever use	10	
Qpmax	Max quantizer	Defines the maximum quantizer that x264 can use	51	
Qpstep	Max QP step	Sets the maximum change in quantizer between two frames	4	
Ratetol		Allowed variance of average bitrate. Only applicable to 1pass		Sets the percentage that x264 can miss the target average bitrate
Ipratio		Sets the target average increase in bitrate for I-frames as compared to P-frames	1.40	
Pbratio		Sets the targest average reduction in bitrate for B-frames as compared to P-frames.	1.30	
chroma-qp-offset		QP difference between chroma and luma. (x264 encodes video as YV12)	0	
Aq-mode	Adaptive Quantization Mode		1	
aq-strength	Adaptive quantize strength	Sets the strength of AQ bias towards low detail ('flat') macroblocks.	1.0	
pass		It controls what x264 will do with the stats file (1,2,3)	Not set	
stats		1st pass stat file name		
rceq	Rate control equation	Don't change		
qcomp	Quantizer curve compression factor	0.0 => Constant Bitrate, 1.0 => Constant Quantizer.	0.60	
cplxblur		Apply a gaussian blur with the given radius to the quantizer curve.	20	
qblur		Apply a gaussian blur with the given radius to the quantizer curve, after curve compression	0.5	

X264 parameter	acronym	Explanation	Default value	recommended
zones				
Qpfile				
ANALYSIS				
partitions		Inter/intra partitions: you enable individual partitions. Partitions are enabled per-frametype (ie I, P, B)	p8x8,b8x8,i8x8,i4x4	
direct		B frames spatial or temporal prediction mode. Recommended: auto	spatial	auto
Direct-8x8		Set the size of the direct prediction mode	-1	
weightb		Allows non-symmetric weighting between references in B-frames	Not set	
me	Motion estimation	Motion estimation methods: Dia (diamond) Hex (hexagon) Umh (uneven mutli-hex) Esa (exhaustive) Tesa (transformed exhaustive)	hex	
Fpel-cmp				
merange	Motion estimation range	max range of the motion estimation search	16	
mvrange	Motion vector range	Set the maximum range of any one motion vector	511.75 (standard)	
subme	Subpixel motion estimation	Set the subpixel estimation complexity: 1. QPel SAD 1 iteration 2. QPel SATD 2 iterations 3. HPel on MB then QPel 4. Always QPel 5. Multi QPel + bime 6. RD on I/P frames 7. RD on all frames 8. RD refinement on I/P frames 9. RD refinement on all frames 10. QP-RD (requires --trellis=2, --aq-mode > 0)	6	
psy-rd		First number is the strength of Psy-RDO to use (requires subme>=6 to activate). The second number is the strength of Psy-Trellis	1.0:0.0	
mixed-refs		Allow references to be selected on a per-8x8 partition, rather than per-macroblock basis	Not set	
no-chroma-me		This disables chroma motion estimation	No	
8x8dct		Adaptive 8x8 DCT enables the intelligent adaptive use of 8x8 transforms in I-frames.	Not set	
trellis		Performs Trellis quantisation to increase efficiency.	Not set	
no-fast-pskip		Disables early skip detection on P-frames.	Not set	

X264 parameter	acronym	Explanation	Default value	recommended
no-dct-decimate		Disables coefficient thresholding on P frames.	Not set	
nr	Noise reduction	Performs fast noise reduction. Recomanded: Default or (100 to 1000 for denoising)	Not set	
deadzone-inter		Set the size of the inter / intra luma quantization deadzone	Not set	
deadzone-intra		Set the size of the inter / intra luma quantization deadzone	Not set	
cqm	Custom quantize matrix	Sets custom quantization matricies from the built in presets of flat or jvt.	Not set	
cqpfile		Custom quantize matric file	Not set	
Cqm4/cqm8			Not set	
INPUT/OUTPUT				
output		Specifies output file name.	Not set	
sar		Specifies the Sample Aspect Ratio in for form of width:height	Not set	
fps		Specifies video framerate.	Not set	
seek		Specified the first frame to encode		
frames		Specifies the maximum number of frames to encode	Not set	
level		Sets the level flag in the output bitstream. Permissible levels include 1, 1b, 1.1, 1.2, 1.3, 2, 2.1, 2.2, 3, 3.1, 3.2, 4, 4.1, 4.2, 5, 5.1	Autodetect	4.1 max
Verbose				
Progress				
Quiet				
No-psnr		Disable the PSNR computation		
No-ssim		Disable the SSIM computation		
threads		enables parallel encoding by using more than 1 thread to increase speed on multi-core systems. Macrobloc rows splitting.	1.5 * logical CPUs detected	auto
Thread-input		Decodes the input video in a separate thread to the encoding process.	Not-set	
Non-deterministic		Slightly improve quality of SMP, at the cost of repeatability	Not-set	
No-asm		Disables all CPU optimisations.	Not-set	
visualize		Enables Macroblock Type visulisations over the encoded video.	Not-set	
Sps-id		Set SPS (Sequence Parameter Set) and PPS (Picture Parameter Set) id numbers	Not-set	
aud		Use access unit delimiters.	Not-set	

Default presets :

The default preset is medium, which is what the other presets are compared against. Table 8 depicts the x264 preset parameters.
* - are differing values from medium.
[ ] values are either set by default, or by another argument.
+ - the x264 argument is used
- - the x264 argument is not used.

Table 8 : x264 preset parameters

Option	ultrafast	superfast	veryfast	faster	fast	medium	slow	slower	veryslow	placebo
no-8x8dct	+*	-	-	-	-	-	-	-	-	-
aq-mode	0*	1	1	1	1	1	1	1	1	1
b-adapt	0*	1	1	1	1	1	2*	2*	2*	2*
bframes	0*	3	3	3	3	3	3	3	8*	16*
deblock	[0:0:0]*	[1:0:0]	[1:0:0]	[1:0:0]	[1:0:0]	[1:0:0]	[1:0:0]	[1:0:0]	[1:0:0]	[1:0:0]
direct	spatial	spatial	spatial	spatial	spatial	spatial	auto*	auto*	auto*	auto*
me	dia*	dia*	hex	hex	hex	hex	umh*	umh*	umh*	tesa*
merange	16	16	16	16	16	16	16	16	24*	24*
no-cabac	+*	-	-	-	-	-	-	-	-	-
no-deblock	+*	-	-	-	-	-	-	-	-	-
no-mbtree	+*	+*	-	-	-	-	-	-	-	-
no-mixed-refs	+*	+*	+*	+*	-	-	-	-	-	-
no-weightb	+*	-	-	-	-	-	-	-	-	-
partitions	none*	i8x8,i4x4*	p8x8,b8x8,i8x8,i4x4	p8x8,b8x8,i8x8,i4x4	p8x8,b8x8,i8x8,i4x4	p8x8,b8x8,i8x8,i4x4	all*	all*	all*	all*
rc-lookahead	0*	0*	10*	20*	30*	40	50*	60*	60*	60*
ref	1*	1*	1*	2*	2*	3	5*	8*	16*	16*
scenecut	0*	40	40	40	40	40	40	40	40	40
subme	0*	1*	2*	4*	6*	7	8*	9*	10*	11*
trellis	0*	0*	0*	1	1	1	1	2*	2*	2*
weightp	0*	1*	1*	1*	1*	2	2	2	2	2

Table 9 shows the encoding settings that remain the same in all presets with x264.

Table 9: Default parameter values in all presets with x264

Option	default
aq-strength	1.00
b-pyramid	2
chroma-qp-offset	-2
crf	23.0
ipratio	1.40
keyint	250
min-keyint	25
psy-rd	1.00:0.00
qcomp	0.60
qpmax	69
qpmin	0
qpstep	4

The Table 10 shows the detailed parameters of the x265 encoder.

Table 10 : x265 parameters

X265 parameter	acronym	Explanation	Default value
ctu	Maximum CU size (width and height).	The larger the maximum CU size, the more efficiently x265 can encode flat areas of the picture, giving large reductions in bitrate.	64
min-cu-size	Minimum CU size (width and height).	By using 16 or 32 the encoder will not analyze the cost of CUs below that minimum threshold, saving considerable amounts of compute with a predictable increase in bitrate. This setting has a large effect on performance on the faster presets.	8
bframes	B-frames max limit	Maximum number of consecutive b-frames.	4
b-adapt		Set the level of effort in determining B frame placement. Values: 0:none; 1:fast; 2:full(trellis)	2
rc-lookahead		Number of frames for slice-type decision lookahead (a key determining factor for encoder latency).	20
scenecut		How aggressively I-frames need to be inserted. The higher the threshold value, the more aggressive the I-frame placement.	40
ref		Max number of L0 references to be allowed.	3
me	Motion estimation	Motion estimation methods: Dia (diamond) Hex (hexagon) Umh (uneven multi-hex) Star Full	Hex

X265 parameter	acronym	Explanation	Default value
merange	Motion estimation range	max range of the motion estimation search	57
subme	Subpixel motion estimation	Amount of subpel refinement to perform. The higher the number the more subpel iterations and steps are performed. Values: 0…7	2
rect		Enable analysis of rectangular motion partitions Nx2N and 2NxN (50/50 splits, two directions). Values: 0 (disabled), 1 (enabled)	0 (disabled)
amp	Asymmetric motion partitions	Enable analysis of asymmetric motion partitions (75/25 splits, four directions). Values: 0 (disabled), 1 (enabled)	0 (disabled)
max-merge		Maximum number of neighbor (spatial and temporal) candidate blocks that the encoder may consider for merging motion predictions. Values : 1..5	2
early-skip		Measure full CU size (2Nx2N) merge candidates first; if no residual is found the analysis is short circuited. Values: 0 (disabled), 1 (enabled)	0 (disabled)
fast-intra		Perform an initial scan of every fifth intra angular mode, then check modes +/- 2 distance from the best mode, then +/- 1 distance from the best mode, effectively performing a gradient descent. Values: 0 (disabled), 1 (enabled)	0 (disabled)
b-intra		Enables the evaluation of intra modes in B slices. Values: 0 (disabled), 1 (enabled)	0 (disabled)
sao	Sample Adaptive Offset loop filter	Toggle Sample Adaptive Offset loop filter Values: 0 (disabled), 1 (enabled)	1 (enabled)
signhide		Hide sign bit of one coeff per TU (rdo). Values: 0 (disabled), 1 (enabled)	1 (enabled)
weightp	Weighted prediction in P slices	Enable weighted prediction in P slices Values: 0 (disabled), 1 (enabled)	1 (enabled)
weightb	Weighted prediction in B slices	Enable weighted prediction in B slices Values: 0 (disabled), 1 (enabled)	0 (disabled)
aq-mode	Adaptive Quantization operating mode	Set Adaptive Quantization operating mode. Values : 0 (disabled) 1 (AQ enabled) 2 (AQ enabled with auto-variance) 3 (AQ enabled with auto-variance and bias to dark scenes)	1 (AQ enabled)
cuTree		Enable the use of lookahead's lowres motion vector fields to determine the amount of reuse of each block to tune	1 (enabled)

X265 parameter	acronym	Explanation	Default value
		adaptive quantization factors. Values: 0 (disabled), 1 (enabled)	
rdLevel		Level of RDO in mode decision. Values : 0…6	3
rdoq-level		Specify the amount of rate-distortion analysis to use within quantization. Values : 0|1|2	0
tu-intra	The transform unit (residual) quad-tree intra depth	This setting limits the number of extra recursion depth which can be attempted for intra coded units. Default: 1, which means the residual quad-tree is always at the same depth as the coded unit quad-tree Values : 0…4	1
tu-inter	The transform unit (residual) quad-tree inter depth	This setting limits the number of extra recursion depth which can be attempted for inter coded units. Values : 0…4	1
aq-strength	Adaptive quantize strength	Sets the strength of AQ bias towards low detail ('flat') macroblocks.	
b-pyramid	B-references insertion	Allow the use of B-frames as references for other frames .	enabled
crf	Quality-controlled variable bitrate	CRF is the default rate control method; it does not try to reach any particular bitrate target, instead it tries to achieve a given uniform quality and the size of the bitstream is determined by the complexity of the source video. The higher the rate factor the higher the quantization and the lower the quality.	28
ipratio		QP ratio factor between I and P slices.	1.4
keyint		Max intra period in frames	250
min-keyint		Minimum GOP size	25
psy-rd		Adds an extra cost to reconstructed blocks which do not match the visual energy of the source block. The higher the strength of --psy-rd the more strongly it will favor similar energy over blur and the more aggressively it will ignore rate distortion.	0
qcomp		Sets the quantizer curve compression factor	0.60
qpmax	Max quantizer	Defines the maximum quantizer that x265 will use	51
qpmin	Min quantizer	Defines the minimum quantizer that x265 will use	0
qpstep	Max QP step	Sets the maximum change in quantizer between two frames	4

Default presets :
The default preset is medium, which is what the other presets are compared against. Table 11 depicts the x265 preset parameters.
* - are differing values from medium.

Table 11 : x265 preset parameters

Option	ultrafast	superfast	veryfast	faster	fast	medium	slow	slower	veryslow	placebo
ctu	32*	32*	32*	64	64	64	64	64	64	64
min-cu-size	16*	8	8	8	8	8	8	8	8	8
bframes	3*	3*	4	4	4	4	4	8	8	8
b-adapt	0*	0*	0*	0*	0*	2	2	2	2	2
rc-lookahead	5*	10*	15*	15*	15*	20	25*	30*	40*	60*
scenecut	0*	40	40	40	40	40	40	40	40	40
ref	1*	1*	1*	1*	2*	3	3	3	5*	5*
me	dia*	hex	hex	hex	hex	hex	star*	star*	star*	star*
merange	57	57	57	57	57	57	57	57	57	92*
subme	0*	1*	1*	2	2	2	3*	3*	4*	5*
rect	0	0	0	0	0	0	1*	1*	1*	1*
amp	0	0	0	0	0	0	0	1*	1*	1*
max-merge	2	2	2	2	2	2	3*	3*	4*	5*
early-skip	1*	1*	1*	1*	0	0	0	0	0	0
fast-intra	1*	1*	1*	1*	1*	0	0	0	0	0
b-intra	0	0	0	0	0	0	0	1*	1*	1*
sao	0*	0*	1	1	1	1	1	1	1	1
signhide	0*	1	1	1	1	1	1	1	1	1
weightp	0*	0*	1	1	1	1	1	1	1	1
weightb	0	0	0	0	0	0	0	1*	1*	1*
aq-mode	0*	0*	1	1	1	1	1	1	1	1
cuTree	0*	0*	0*	0*	1	1	1	1	1	1
rdLevel	2*	2*	2*	2*	2*	3	4*	6*	6*	6*
rdoq-level	0	0	0	0	0	0	2*	2*	2*	2*
tu-intra	1	1	1	1	1	1	1	2*	3*	4*
tu-inter	1	1	1	1	1	1	1	2*	3*	4*

Table 12 shows the encoding settings that remain the same in all presets with x265.

Table 12 : Default parameter values in all presets with x265

Option	default
aq-strength	0.00
b-pyramid	enabled
crf	28.0
ipratio	1.40
keyint	250
min-keyint	25
psy-rd	1.00:0.00
qcomp	0.60
qpmax	51
qpmin	0
qpstep	4