Power-Aware HEVC Decoding with Tunable Image Quality

Erwan Nogues*, Simon Holmbacka†, Maxime Pelcat*, Daniel Menard* and Johan Lilius†
*UMR CNRS 6164 IETR Image Group, INSA de Rennes
Email: {erwan.nogues,maxime.pelcat,daniel.menard}@insa-rennes.fr
†Department of Information Technologies, Åbo Akademi University, FIN-20520 Turku
Email: {sholmbac,jolilius}@abo.fi

***Abstract*—A high pressure is put on mobile devices to support increasingly advanced applications requiring more processing capabilities. Among those, the emerging High Efficiency Video Coding (HEVC) provides a better video quality for the same bit rate than the previous H.264 standard. A limitation in the usability of a mobile video playing device is the lack of support for guaranteeing stand-by time and up time for battery driven devices. The Green Metadata initiative within the MPEG standard was launched to address the power saving issues of the decoder and defines the technology requirements. In this paper, we propose a HEVC decoder with tunable decoding quality levels for maximum power savings as suggested in the scope of the Green Metadata initiative. Our experiments reveal that the modified HEVC video decoder can save up to 28 % of power consumption in real-world platforms while keeping better quality than decoding with H.264.**

I. Introduction

Smart phones, tablets and media players are the major consumers of multimedia content. In [1], it is reported that video on mobile devices is expected to exceed 70 % of the Internet traffic in 2016. Smart management of the device and its use of energy is therefore crucial in order to support new features without altering the usability. Acknowledging that power consumption is a crucial problem on mobile devices, MPEG launched an ad-hoc working group also called Green Metadata [2] to reduce the power consumption of the video processing. Among the general requirements of the Green Metadata, a recommendation is that the decoder shall offer the means to compromise between the quality of the video and its power consumption.

The High Efficiency Video Coding (HEVC) is the new MPEG standard for video compression and provides the same video quality at half the bit rate compared to the previous standard H.264/AVC. Bossen et al. show in [3] that the complexity of the HEVC decoder is similar to the one of H.264/AVC which means that the end-user can benefit from improved video quality with no additional cost on processing time. The feature can be exploited for low power tunable video processing and power can be saved by scaling down the hardware resources without quality distortion.

The general solution to reduce power consumption is to enable clock frequency reduction while keeping performance guarantees. Power saving techniques such as DVFS (Dynamic Voltage and Frequency Scaling) can be utilized to bring the CPU into the most power efficient state, this state depending on the system workload. This technique enables the reduction of the processor power consumption by providing only the necessary power to execute a job. Techniques such as DVFS can directly be utilized in combination with tunable image qualities to increase power savings in HEVC decoders.

In this paper, we show how power consumption of a HEVC video decoder can be reduced by providing tunable video quality. The tuning functionality is based on dynamic activation of in-loop filters and on dynamic activation of the interpolation filters, reducing the complexity of the decoder. We show that a good compromise between image quality and power consumption can be achieved by decreasing the filter complexity while still maintaining a higher decoding quality than H.264/AVC. Finally we can demonstrate that the modified HEVC decoder uses less power than the reference implementation with a power gain of up to 28% on a real hardware platforms without changing the initial HEVC bitstream and by using the standard GCC compiler and a unmodified Linux OS. We also show that the suggested filtering techniques result in similar power savings on both embedded ARM embedded platforms and on Intel desktop platforms.

The rest of the paper is organized as follows: Section II presents the related work. Section III introduces the proposed method and its impact on quality with respect to the H.264/AVC standard. The proposed method takes the H.264/AVC as the lower bound for rate-distortion curves. Section IV presents our experimental results for power optimization on a hardware platform and conclusions are given in Section V.

II. Related work

The present study addresses the power consumption of video decoding. Various techniques have been studied in the past at both application level [4] and architecture level [5].

The work in [6] formulated a rigorous scheduling and DVFS policy for slice-parallel video decoders on multi-core hardware with QoS guarantees on the playback. The authors presented a two-level scheduler which firstly selects the scheduling and DVFS utilization per frame and secondly maps frames to processors and set their clock frequencies. In our work, we move the abstraction of the problem to decoder optimization where the decoder changes the functional blocks call to reduce the decoding complexity. The system can reduce its operating frequency to reduce the power consumption with already existing techniques and implementations.

Performance optimization can be also done by scalable mechanisms. The Scalable High efficiency Video Coding (SHVC) standard is the scalable extension of the High Efficiency Video Coding (HEVC) standard [7]. The SHVC standard aims to provide spatial and quality scalability with a simple and efficient coding architecture. In [8], an implementation of a SHVC decoder is presented with performance comparison. The decoder can control its video quality according to the number of decoded layers. This mechanism can be used by the decoder to adapt the number of decoded layers to its own power capabilities. It results in finding a trade-off between the quality and the decoding speed. In [8], there is also a complexity comparison between the HEVC decoder and the SHVC. It is noted that HEVC decoding can be twice faster than SHVC decoding. In our approach, we look for the best compromise between quality and decoding speed but with no major additional complexity on decoder side.

Another technique is proposed in [9] by He et al. for mobile HEVC streaming. Their purpose is to define a power-aware system in which the decoder could feedback its power level to the encoder. This work follows one of the requirements of Green Metadata [2]. The encoder would in this case adapt, by segments (e.g. 5 seconds), the content of the bitstream to reduce the decoding complexity. The main advantage of this approach is that there is no added complexity on decoder side. However, it creates a unique link between the encoder and the decoder. To support such method, a specific stream per decoder needs to be set up which can have a significant impact on the network load when the number of decoders grows. This method is also not suitable for broadcast systems. In our work we propose to only manage the power reduction on the decoder side. The main advantage is that the encoder does not have to handle different decoder implementations or individual power and performance requirements.

In [1], the analysis of power consumption on a smart phone reveals that the display on the screen consumes the largest part of the total power. Indeed, 400 mW is needed for the display, 300mW for the video decoding, 250 mW for the idle part and 300 mW for downloading the video. Chang et al. [10] propose back light scaling for LCD system. The induced distortion is compensated by an appropriate image mechanism to keep as close as possible the perceived image contrast. Shin et al. [11] propose a new principle for the OLED technology adopted in newer equipments. Power consumption is then improved but it highly relies on the hardware technology used by the end device. The decoded video has still good performance but is not exactly compliant with the HEVC reference output. In our approach, we also acknowledge that power reduction can be done at a cost of a slight modification in the video decoding but our method is not linked to the hardware characteristics of the device.

III. Proposed methods to tune power consumption of HEVC decoders

The primary decoder modifications consist of activating different Finite Impulse Response (FIR) filters according to a tunable input parameter called *Activation level*. In this section, we analyse how these filter modifications can provide a fine grain tunable parameter for complexity and we compare the decoded video quality with the H.264/AVC.

A. Modified HEVC decoder with Multiple Activation Levels of the filters

We use a standard structure of the HEVC decoder. It is split into several blocks as shown in Figure 1. In the first step, the entropy decoder extracts the different syntax elements from the video stream using arithmetic coding after which the residual data are dequantized and transformed using an inverse Discrete Cosine Transform (DCT) process. The prediction of the frames is then applied, and can be either of intra- or inter-frame type depending on the input bitstream parameters. In the case of the inter-frame prediction, a prediction is computed based on the previously decoded pictures which estimates the motion vectors at a fractional pixel level. Finally the Deblocking Filter (DF) and Sample-Adaptive Offset filter (SAO) are applied on the reconstructed data to reduce potential artifacts and increase the picture quality.

Power reduction can be achieved in various ways, especially if the quality is allowed to be degraded. This statement is used for the complexity reduction. To continue benefiting from the HEVC improvements with respect to H.264/AVC, the maximum quality distortion is set to the one from H.264/AVC.

The HEVC decoding process has been profiled in [3], [12] on various platforms such General Purpose Processor (GPP), Digital Signal Processor (DSP) and with different types of encodings such as Random Access (RA) and All Intra (AI) for different levels of compression and use cases. RA configurations are used typically for broadcasting, and use a pyramidal structure for picture reordering. The reference image is sent periodically and all other frames are deduced from each other with the inter-frame prediction. In AI, all pictures use I-slices for encoding and only intra-frame prediction. It is explained in [3] that the Motion Compensation (MC), DF and the SAO utilize roughly 43%, 17% and 4% on RA profile, and 0%, 13% and 6% of processing time on AI profiles. The implementation of the used reference HEVC [13] reveals similar results. Based on the high relative complexity of these functions, the proposed modified HEVC focuses on them to reduce the power consumption. The modifications are illustrated in Figure 1 as grey rectangles in both the In-loop filtering and the Motion compensation part of the decoder. The following sections present details regarding the modifications of a reference HEVC decoder implementation.

In-loop filtering

The DF and SAO filters are grouped into a block called in-loop filters shown in Figure 1, which can be applied sequentially to the reconstructed picture. DF filter aims at reducing the blocking artifacts as a result of block-based coding. DF filter is similar to the filter used in H.264/AVC whereas SAO filter is new in HEVC. SAO filter processing is done after the application of the DF filter to provide additional refinement of the reconstructed video. It can enhance the video representation in both smooth areas and around edges [7]. The complexity and the performance of DF is reported in details in [14], and it is shown that complexity and performance were improved when changing from H.264/AVC to HEVC. By removing the HEVC in-loop filters, the decoding complexity is reduced and can be exploited for power reduction. Section IV reports the power consumption for different levels of the filter activations.

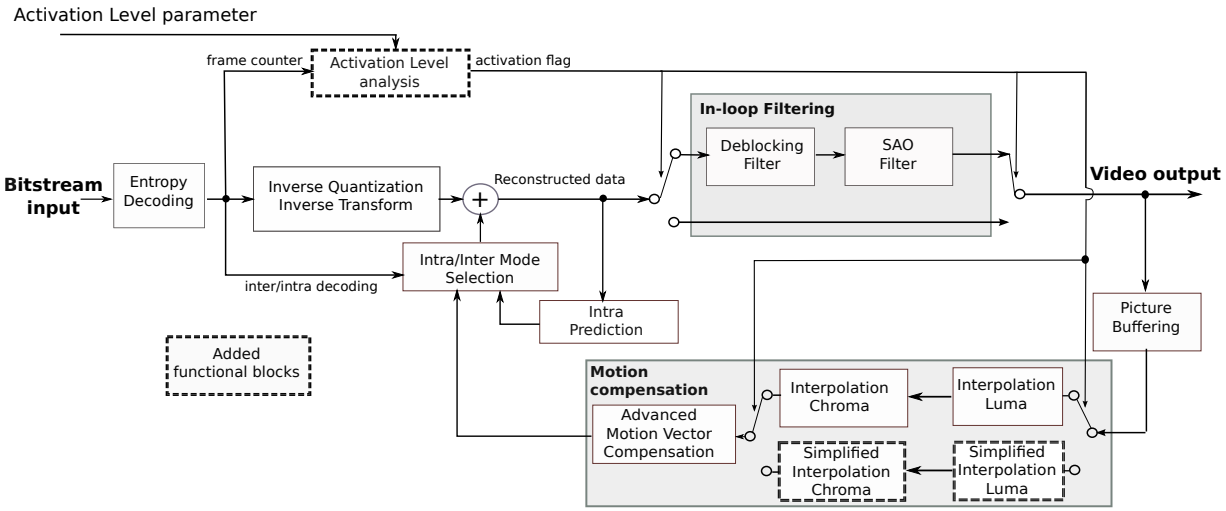


Figure 1. Block diagram of the modified HEVC decoder. Dashed blocks are the added blocks. Grey rectangles indicate where the filtering is modified

The quality distortion is also expected to be small compared to H.264/AVC and is reported in section III-B.

Motion compensation filtering

The second modification is on the motion compensation (MC) and is used to simplify some FIR filters to reduce the HEVC decoder complexity. This section describes how to reach this claim by using a method to reduce the number of taps in the FIR filters.

For fractional motion vector compensation, 1-D interpolation filters are used in HEVC [7]. The luma part is constructed of two different types of filters: a 8-tap filter for half-pel positions and a 7-tap asymmetric filter for quarter-pel positions. The chroma part simply uses a 4-tap filter. They are all implemented with FIR filters. To reduce their complexity, the proposed method uses a smaller number of taps. The filter size is set to 3 for luma and 1 for chroma. It implies that new filter taps need to be synthesized and the same filter synthesis method is used as during HEVC standardization.

The interpolation process uses a DCT transform for the filter synthesis. Assuming a local list of pixels $\{p_i\}(i = M_{min}, .., M_{max})$ of $Size = M_{max} - M_{min} + 1$, the forward DCT generates the Fourier coefficient C_k (Eq. 1). The pair of forward-inverse transforms can be pre-calculated and merged for fractional position [15].

$$C_k = \frac{2}{Size} \cdot \sum_{l=M_{min}}^{M_{max}} p(l) cos(\frac{(2 \cdot l - 2 + Size) \cdot k \cdot \pi}{2 \cdot Size}) \quad (1)$$

In HEVC [7], the 8-tap filter designed for the luma is using for example $M_{min} = -3$, M_{max} =4 and $Size = 8$. As stated before, to reduce the complexity, the proposed method sets the $Size$ parameter to 3 instead of 8 in Eq. 1. As a consequence, M_{min} is equal to -1 and M_{max} is equal to 1. Finally, for the fixed point implementation, a scaling factor of 2^s where s is 6 is used to multiply the floating taps and round them to the nearest integer. The Tables I and II describe the original and modified filters for all the interpolation factors α standardized in HEVC. The original filter taps correspond to the HEVC implementation [15] and the modified filter taps correspond the proposed method to reduce the complexity.

Table I. LUMA INTERPOLATION FILTER : ORIGINAL AND MODIFIED

α	Original $filter(\alpha)$	Modified $filter(\alpha)$
1/4	(-1, 4, -10, 58, 17, -5, 1)	(-7, 58, 13)
1/2	(-1, 4, -11, 40, 40, -11, 4, -1)	(-9, 41, 32)

Table II. CHROMA INTERPOLATION FILTER : ORIGINAL AND MODIFIED

α	Original $filter(\alpha)$	Modified $filter(\alpha)$
1/8	(-2, 58, 10, -2)	(64)
1/4	(-4, 54, 16, -2)	(64)
3/8	(-6, 46, 28, -4)	(64)
1/2	(-4, 36, 36, -4)	(64)

Dynamic filtering - $ActivationLevel$ definition

Our primary decoder modifications consist of simplifying the filters present in the in-loop filtering and motion compensation blocks of figure 1. The decoding complexity is reduced as less operations are needed but it results in a quality distortion. In this section, we describe how the proposed modifications can be done to offer a fine grain level of quality tuning. To be able to tune the quality distortion, the modifications of the filters are not applied on all the frames. A decision is taken at a frame level to decide if the modification of the DF and MC filters shall be applied to the current frame. When the filters are modified as per Section III-A at every frame, a distortion of 1.2 dB of the Peak Signal-to-Noise Ratio (PSNR) (Figure 2) is observed on a HD video. A tunable parameter called $ActivationLevel$ is introduced to leverage the distortion. Twelve steps of $ActivationLevel$ are defined to propose a maximum of 0.1 dB of distortion per step. By setting $ActivationLevel$ $\{0..12\}$, the decoder can dynamically use the filters to be either equivalent to HEVC ($ActivationLevel = 0$ – never change the filters, no power optimization and no quality distortion) or highly modified ($ActivationLevel = 12$ – use the modified filters on all the frames, power optimization to the maximum and maximum quality distortion). In other words, the modified

HEVC decoder is fully backward compatible with HEVC if $ActivationLevel = 0$.
An extra functional block called *ActivationLevel analysis* (Figure 1) is added to decide when the modifications of Section III-A shall apply with the frame number as a input. Table III summarizes the frame number when the modifications apply.

Table III. MODIFIED HEVC *ActivationLevel* TABLE

ActivationLevel	Frame number index {0,..,12}
0	*never activated - legacy HEVC*
1	(0)
2	(0, 6)
3	(0, 4, 8)
4	(0, 3, 6, 9)
5	(1, 3, 7, 9, 11)
6	(1, 3, 5, 7, 9, 11)
7	(0, 2, 4, 5, 6, 8, 10)
8	(1, 2, 4, 5, 7, 8, 10, 11)
9	(1, 2 ,3, 5, 6, 7, 9, 10, 11)
10	(1, 2 ,3, 4, 5, 7, 9, 10, 11, 12)
11	(0, 1, 2 ,3, 4, 5, 7, 8, 9, 10, 11)
12	*new blocks always activated*

B. Performance assessment - Comparison to H.264/AVC

The rate-distortion is used as the evaluation metric for the decoder. As described in Section III-A, the HEVC filters are only activated on precomputed frame numbers according to the $ActivationLevel$ parameter, which causes quality distortion of the decoded video. In this section the video quality distortion is evaluated according to the $ActivationLevel$ parameter. Ohm et al. presented in [16] a survey of the HEVC performance versus previous video standards. PSNR is used as the distortion metric in our quality measurements as in [16]. The metric is a combined PSNR of the luma (Y) and the chroma (U,V) components per image with different weights, $PSNR_{YUV}$.

$$PSNR_{YUV} = (6 \cdot PSNR_Y + PSNR_U + PSNR_V)/8, \quad (2)$$

where $PSNR_Y$, $PSNR_U$ and $PSNR_V$ are independently computed as follows:

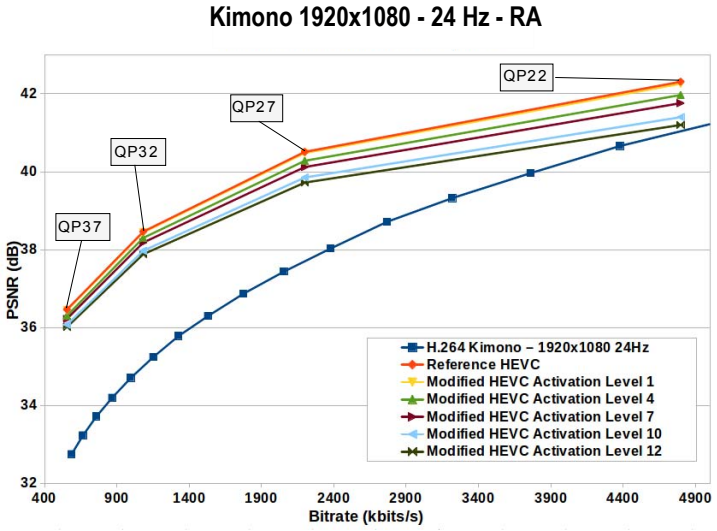
$$PSNR = 10 \cdot log_{10}(d^2/MSE), \quad (3)$$

where d is 255, MSE is the Mean Square Error of the reference image to the decoded image. The PSNR of the video is computed by averaging the PSNR per image.

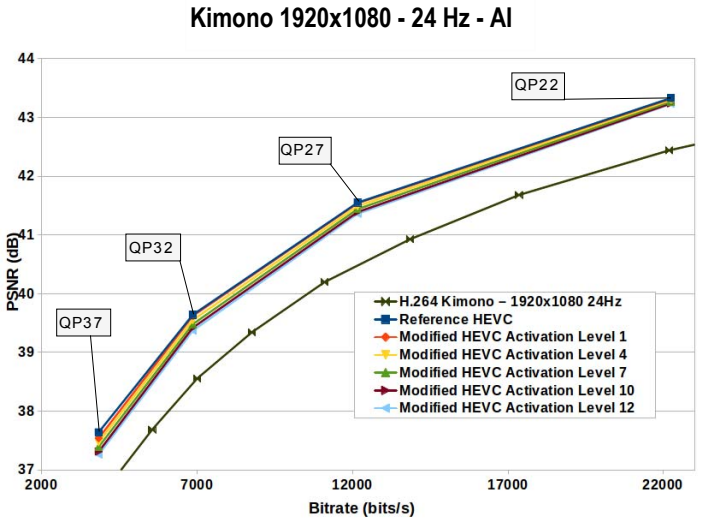
Our HEVC decoder is based on *OpenHEVC* [13] and the input test sequences from the JCT-VC common test are used. For H.264/AVC, the JM reference software has been used [17]. Each test sequence is coded into twelve different bit rates. The quantization parameter QP_i varies in the range of 20 to 42 with the same methods described in [16].

A Class B (1920 x 1080 pixels) video called *Kimono* is selected as it is commonly used for performance evaluation [12], [16] and, for each bitstream, the RA and the AI profiles are evaluated to test various implementations of the in-loop filtering and the motion estimation in practice. The rate-distortion curves of the quality evaluation are shown in Figure 2 for different $ActivationLevel$ values.

The bit-rate achievements of the reference HEVC and H.264/AVC are similar to the results presented in [12], [16]. The proposed method of modified HEVC presents intermediate results for distortion levels. For AI profile, the distortion is lower than 0.4 dB and the decoder can still benefit from the HEVC's superior performance. For RA profile, the modified HEVC can still benefit from the higher performance of HEVC at low bit rate, and the performance depends on the complexity level at higher bit rate. As seen in Figure 2, the performance is *at least* better than the H.264/AVC decoder for all test cases. It can be noted that the $ActivationLevel$ parameter provides fine grain performance decoder with less than 0.1 dB per step. As a conclusion, the proposed decoder can be tuned with different levels of quality and outperforms the H.264/AVC on the rate-distortion curves.



(a)



(b)

Figure 2. Distortion comparison between the HEVC, Modified HEVC and H.264/AVC for RA profile *(a)* and AI profile *(b)*

IV. POWER MEASUREMENTS

The second set of benchmarks were conducted to evaluate the power savings of our modified HEVC decoder. As a starting point we used a ready-for-execution reference software [12], which was modified with the functionalities presented in the previous sections. While we acknowledge that more optimized versions of the decoder exist [3], our intentions are

to compare the legacy implementation to our modified decoder in terms of power savings on general purpose hardware.

The power measurements were conducted on two different hardware platforms. Firstly we used an octa-core Exynos 5410 SoC based on the big.LITTLE configuration with four ARM Cortex-A15 cores and four ARM Cortex-A7 cores. This SoC is widely used in recent smart phones and tablets [18]. The CPU has a maximum clock frequency of 1600 MHz and can be frequency scaled down to 250 MHz. Our software was run on top of a default Linux kernel which uses an automatic CPU cluster switching from the energy efficient A7s to the powerful A15s as the clock frequency switches between 600 and 800 MHz. This means that either the A7s or the A15s can be active at the same time.

Secondly we used a quad-core desktop CPU based on the Intel i7-3770 with a clock frequency range between 1.6 GHz and 3.4 GHz. The hyperthreading and the Intel Turbo Boost was disabled for all experiments. We used a standard Linux kernel and no modifications were made to the default power management system, and the *ondemand* [19] frequency governor was used in all experiments on both platforms.

The power measurements were obtained by running the HEVC decoder on four threads for a fixed number of frames and with various configurations. The power was read from internal power registers on the ARM platform, and from an external power meter directly connected to the current feed of the CPU on the i7 platform. All power readings were obtained with an accuracy of four decimals and the readings were stored with a sampling period of 100 ms. Listing 1 outlines the pseudo code for the power measurements using a shell script:

```
loop over parameters{
  start_power_reading()
  start_HEVC() <parameters> <video>
  stop_power_reading()
  store_reading() }
```

Listing 1. Pseudo code for power measurements

We used the same 1080p video as in Section III-B and the following parameters were used in the experiments:

- QP: [22, 27, 32, 37]
- Frame type: [AI frames, RA frames]
- Filter *ActivationLevel*: [0, 1, 4, 7, 10, 12]

Each decoding run was iterated 10 times for increased accuracy and, with the exception of minor Linux background tasks, the CPU did only execute the decoder during all tests. Table IV shows the average power consumption for the ARM platform and Table V shows the average power consumption for the Intel platform. Table VI furthermore shows the average standard deviation for each complexity level on both platforms, from which it can be noted that the standard deviation is not impacted by the $ActivationLevel$ and stay stable on both platforms.

As seen in the Tables IV and V the power consumption can be reduced by setting the filter $ActivationLevel$. The experimental results show that a similar trend is seen on ARM and Intel platforms even though they are not intended for the same use.

Table IV. POWER (IN WATTS) MEASUREMENTS OF ARM PLATFORM

Sequence RA	Legacy	Level1	Level4	Level7	Level10	Level12
Kimono QP22	4.168	4.124	3.950	3.621	3.512	3.325
Kimono QP27	3.773	3.724	3.398	3.216	2.949	2.798
Kimono QP32	3.351	3.329	2.978	2.828	2.586	2.448
Kimono QP37	3.073	3.014	2.748	2.507	2.320	2.185
Sequence AI	Legacy	Level1	Level4	Level7	Level10	Level12
Kimono QP22	5.149	5.334	4.978	4.670	4.518	4.409
Kimono QP27	4.526	4.550	4.248	4.027	3.777	3.602
Kimono QP32	4.005	3.885	3.654	3.512	3.333	3.163
Kimono QP37	3.387	3.324	3.157	2.992	2.870	2.780

Table V. POWER (IN WATTS) MEASUREMENTS OF INTEL PLATFORM

Sequence RA	Legacy	Level1	Level4	Level7	Level10	Level12
Kimono QP22	18.83	18.69	17.51	16.67	15.72	15.0
Kimono QP27	16.28	16.17	15.37	14.66	13.75	13.27
Kimono QP32	14.91	14.78	14.15	13.48	12.65	12.17
Kimono QP37	13.91	13.74	13.2	12.62	11.92	11.51
Sequence AI	Legacy	Level1	Level4	Level7	Level10	Level12
Kimono QP22	22.46	22.26	21.32	20.47	19.78	19.31
Kimono QP27	19.81	19.45	18.55	17.6	17.17	16.75
Kimono QP32	17.8	17.6	16.72	15.99	15.69	15.06
Kimono QP37	15.55	15.45	15.13	14.74	14.23	13.71

The power saving in percentage is defined as:

$$PowerSaving(\%) = (1 - \frac{Power_{new}}{Power_{reference}}) \cdot 100 \qquad (4)$$

where $Power_{new}$ is the average power of the modified HEVC decoder and $Power_{reference}$ is the average power of the reference implementation.

To save power in the decoder, from a system level perspective, the first option is to reduce the bitrate; for example, by using QP32 RA sequences on the ARM platform saves 19.60% of power compared to using the QP22 sequence, and on the Intel platform, the saving is 20.82%. When correlating these results with the PSNR measurements from Figure 2, this power saving is done at a cost of 3.85 dB. With our proposal, the QP22 sequence could instead be decoded with an $ActivationLevel$ of 12, which leads to a similar power saving of 22.02% on the ARM platform and 20.34% on the Intel platform. In this case, the quality distortion is only 1.11 dB. When using the AI profile, QP27 sequence saves 12.09% compared to QP22 sequence on the ARM platform, and 10.54 % on the Intel platform, and with a quality distortion of 1.78 dB. By using our decoder and a QP22 bitstream, similar power savings can be achieved with $ActivationLevel$ of 10. The resulting quality distortion is only 0.09 dB. This means that the proposed method achieves an equal power saving but with a better quality compared to reducing the bitrate on the legacy implementation.

Finally, Figure 3 illustrates the power savings as a function of the PSNR distortion for bitstreams of QP22, QP27, QP32 and QP37 for both RA and AI profiles with ARM and Intel platforms. By utilizing the trade-off between video quality and power savings presented in Figure 3, a power saving scheme can be adopted in the decoder to achieve minimum power consumption with user definable video quality. The decoder

Table VI. STANDARD DEVIATION OF BOTH PLATFORMS

	Legacy	Level1	Level4	Level7	Level10	Level12
ARM RA	0.66	0.63	0.63	0.62	0.59	0.55
ARM AI	0.56	0.50	0.51	0.47	0.44	0.47
Intel RA	0.18	0.17	0.17	0.17	0.17	0.15
Intel AI	0.08	0.08	0.08	0.08	0.09	0.08

device is hence able to adapt its decoding characteristics with its resource requirements at any time. Indeed, with our proposal, the decoder can easily implement its own decoding strategy according to the use case.

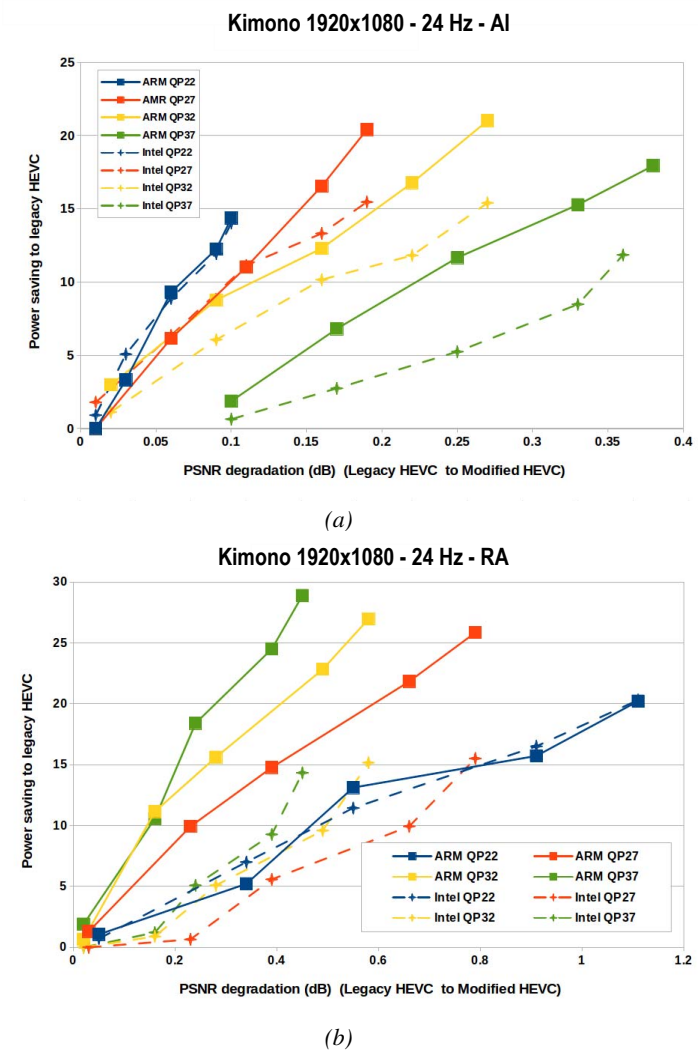


Figure 3. Power savings with RA profile *(a)* and AI profile *(b)* vs. quality distortion for both ARM and Intel platforms compared to legacy implementation

V. Conclusion

We propose in this paper modifications of a HEVC decoder to decrease the power consumption compared to the legacy HEVC. Modifications are made to the in-loop filters and the motion compensation filters to allow tunable video quality; an authorized feature in Green Metadata decoding. The proposed decoder applies modifications on video frames according to an $ActivationLevel$ parameter to tune the power saving and the quality. We show power savings of up to 28 % on real-world platforms while the quality is only slightly degraded but still better to the previous video compression standard H.264/AVC. By using this mechanism, the decoder can adjust its power consumption with an a-priori knowledge of the Quality of Experience of the video display as suggested in MPEG/Green Metadata standard group. In the same fashion, the video quality can also be adjusted to power constraints such as battery lifetime.

VI. Acknowledgment

This work was partially supported by BPI France, Region Ile-de-France, Region Bretagne and Rennes Metropole through the French Project GreenVideo.

References

[1] "Embedding content information in video streams for energy-efficient video processing on mobile devices," *ISO/IEC JTC1/SC29/WG11 MPEG2012/*, April 2012.

[2] "Context, objectives, use cases and requirements for green mpeg," *SO/IEC JTC1/SC29/WG11/N13468*, April 2013.

[3] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "Hevc complexity and implementation analysis," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1685–1696, 2012.

[4] S. Jafri, M. Tajammul, A. Hemani, K. Paul, J. Plosila, and H. Tenhunen, "Energy-aware-task-parallelism for efficient dynamic voltage, and frequency scaling, in cgras," in *Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIII), 2013 International Conference on*, 2013, pp. 104–112.

[5] I. Hong, D. Kirovski, G. Qu, M. Potkonjak, and M. Srivastava, "Power optimization of variable voltage core-based systems," in *Design Automation Conference, 1998. Proceedings*, 1998, pp. 176–181.

[6] N. Mastronarde, K. Kanoun, D. Atienza, P. Frossard, and M. van der Schaar, "Markov decision process based energy-efficient on-line scheduling for slice-parallel video decoders on multicore systems," *Multimedia, IEEE Transactions on*, vol. 15, no. 2, pp. 268–278, 2013.

[7] G. J. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1649–1668, 2012.

[8] W. Hamidouche, M. Raulet, and O. Deforges, "Parallel shvc decoder: Implementation and analysis," *IEEE conference on ICME*, 2014.

[9] Y. He, M. Kunstner, S. Gudumasu, E.-S. Ryu, Y. Ye, and X. Xiu, "Power aware hevc streaming for mobile," in *Visual Communications and Image Processing (VCIP), 2013*. IEEE, 2013, pp. 1–5.

[10] N. Chang, I. Choi, and H. Shim, "Dls: dynamic backlight luminance scaling of liquid crystal display," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 12, no. 8, pp. 837–846, 2004.

[11] D. Shin, Y. Kim, N. Chang, and M. Pedram, "Dynamic voltage scaling of oled displays," in *Design Automation Conference (DAC), 2011 48th ACM/EDAC/IEEE*. IEEE, 2011, pp. 53–58.

[12] M. Chavarrias, F. Pescador, M. Garrido, M. Raulet *et al.*, "A dsp-based hevc decoder implementation using an actor language dataflow model," *Consumer Electronics, IEEE Transactions on*, vol. 59, no. 4, pp. 839–847, 2013.

[13] "The Open HEVC - open source project," https://github.com/OpenHEVC/openHEVC.

[14] A. Norkin, G. Bjontegaard, A. Fuldseth, M. Narroschke, M. Ikeda, K. Andersson, M. Zhou, and G. Van der Auwera, "Hevc deblocking filter," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1746–1754, 2012.

[15] U. Kemal, A. Alshin, E. Alshina, F. Bossen, W. Han, J. Park, and J. Lainema, "Motion compensated prediction and interpolation filter design in h. 265/hevc," 2013.

[16] J. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standardsâĂŤincluding high efficiency video coding (hevc)," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1669–1684, 2012.

[17] "H.264/MPEG-4 AVC Reference software, joint model 18.6," http: iphome.hhi.de/suehring/tml/.

[18] "Samsung GALAXY S4 Teardown," http://www.techinsights.com/inside-samsung-galaxy-s4/, 2013.

[19] D. Brodowski, "Cpu frequency and voltage scaling code in the linux(tm) kernel," https://www.kernel.org/doc/Documentation/cpu-freq/governors.txt, 2013.