Thermal Optimization using Adaptive Approximate Computing for Video Coding

[1,2]Daniel Palomino, [3]Muhammad Shafique, [1]Altamiro Susin, [3]Jörg Henkel
[1]Informatics Institute, PPGC, Federal University of Rio Grande do Sul (UFRGS), Brazil
[2]Center for Engineering, Federal University of Pelotas (UFPel), Brazil
[3]Chair for Embedded Systems (CES), Karlsruhe Institute of Technology (KIT), Germany

***Abstract*— This paper presents a thermal optimization technique that adaptively employs varying degree of approximations at both algorithm and data levels in order to reduce the temperature associated with the high efficiency video coding process while maintaining good quality results. The technique evaluates, at run-time, the regions of a video sequence, frame-by-frame, in terms of tolerance to imprecise computations. It adapts the amount of approximation errors based on the video sequence properties and application-specific knowledge. The proposed technique adaptively controls the strength of approximations (at both algorithm and data levels) depending upon the varying resilience properties of coding different regions with different texture/motion properties. Our content-driven approximate computing technique demonstrates the potential to improve the thermal profile of a chip. Experimental results show that our technique improves temperature profiles by reducing the on-chip temperature by about 10º C on average, while maintaining good quality results.**

I. Introduction and Related Work

Video services have widely spread in the consumer market in the past years. According to [1] video will consume 80%-90% of the global internet traffic by 2017. New features such as higher resolutions (nowadays up to 4K and 8K) and higher immersion provided by 3D technology improve the user experience, but demand for efficient compression techniques. The High Efficiency Video Coding (HEVC) [2] standard has recently been released to provide the high coding efficiency (i.e., low bit rates with high video quality) to fulfil the demands of high-end video contents. However, the computational complexity, and consequently the power consumption and on-chip temperatures, of the video encoding process have increased when comparing with the previous H.264 encoder [3][4][5].

The high complexity of the HEVC can be well complemented by the high integration density due to the continuous shrinking of the transistor feature sizes. This paves the path for the high-performance computing systems with many processing cores, operating at high clock frequencies. However, the expected voltage scaling has hit a roadblock due to the failure of Dennard's scaling [6][7] that has resulted in trends of high power density (power/area), elevated on-chip temperature (i.e., thermal hotspots), and consequently high cooling costs [8]. Furthermore, thermal hotspots negatively affect the reliability and lifetime of devices since most of the aging effects (such as electromigration, NBTI, HCI, and TDDB) are aggravated at high temperatures [9][10]. Therefore, *improved thermal profiles for embedded video processing systems are necessary*, especially when considering the high complexity of the HEVC.

Recently, the concept of *approximate computing* have gathered a lot of attention and have been seen as an attractive way to improve performance or power efficiency by compromising the application quality within the tolerable ranges [11]. The basic idea of approximate computing is to exploit the applications resilience to errors in order to reduce the total amount of computations at the software and/or hardware level. Several studies have applied approximate computing techniques at different layers of the computing stack, for instance, circuit design [12][13], architectures [14][15], and application software [16][17] to reduce power consumption but have *not yet explored their impact on the thermal profile optimization*. Due to inherent resilience of various functional blocks (like motion estimation) and varying levels of user perception, video coding (HEVC in our case) is a well-suited application for approximate computing and it can tolerate a varying degree of errors in the output visual quality after the coding process. Earlier works have explored concepts of curtailing mode computations [18], compute effort scaling [17], and efficient data management [19] to trade computational complexity and power consumption with the output visual quality, respectively. However, *state-of-the-art have not yet explored the potential of approximate computing to alleviate the on-chip thermal profiles* in complex video coding systems equipped with HEVC.

The goal of this paper is to adaptively employ approximate computing depending on the video content properties and the application knowledge in order to explore the tradeoffs between on-chip temperatures, computational complexity and visual quality.

A. State-of-the-Art in Temperature Optimization for Video Coding and Their Limitations

The temperature optimization works usually target at keeping the temperature under safe thermal limits or at lowering the overall chip temperature [19]. When the target is video coding/decoding systems, works such as [21][22] use Dynamic Voltage and Frequency Scaling (DVFS) together with temporal and spatial degradation (by frame drops) as a technique to keep the temperature under the critical temperature threshold. However, these works results in significant quality loss due to frame drops, and primarily target decoder instead of encoder that is more challenging for temperature optimization due to its >10x higher complexity compared to the decoder [23]. The work in [24] selects a video quality degradation mode to compensate for the quality effects of different DTM policies. However, it does not exploit the video content and algorithmic properties to optimize temperature. Also, all these works mainly target old standards such as MPEG2 and H.264 and may not perform efficient for HEVC considering its new coding tools [3]. The work in [5] proposes a thermal management policy that adapts encoder configurations at run-time to meet the temperature constraints. However, changing some of the encoder configurations, such as the Quantization Parameter (QP) can lead to excessive loss in the output visual quality, which may not be a user-specified tolerable range. *Overall, none of these state-of-the-art techniques have explored the potential of different types of approximate computing and their adaptations based on the video content properties to improve the on-chip thermal profiles of video coding systems*.

In summary, there is a need for a temperature optimization technique that improves temperature profiles of video coding systems through adaptively employing varying degree of approximations at both algorithm and data levels.

B. Our Novel Contributions

In order to address the above challenges, we propose the following key contributions:

1) Thermal Optimization using Adaptive Approximate Computing for High Efficiency Video Coding: We propose a content-driven approximate computing technique that adaptively applies varying degree of approximations at both algorithm and data levels in

978-3-9815370-6-2/DATE16/

order to reduce temperature for the High Efficiency Video Coding (HEVC), while maintaining good visual quality results. It classifies different regions in a video sequence considering their texture and motion information in order to determine the potential error resilience and thereby enabling adaptive approximations. A video region with high resilience is more amenable to (aggressive) approximate computing and yet leading to a low degradation in the output quality.

2) Error Tolerance Analysis for Different Videos: For designing and applying our content-driven approximate computing technique, we perform an extensive analysis of error tolerance for different video sequences and even different regions in a video sequence based on their content properties such as motion and texture. Our analysis shows that regions with different resilience properties can support different degrees of approximations in order to reduce the total workload associated with the HEVC encoding process and thereby leading to lower on-chip temperatures. Furthermore, it is possible to maintain good video quality results when the approximations are performed carefully in appropriate regions considering their resilience properties.

Paper Organization: The rest of this paper is organized as follows: Section II presents our error tolerance analysis of HEVC. Section III presents our adaptive approximate computing technique for temperature optimization. Section IV presents the results of our technique in terms of video quality and temperature, and also a comparison with state-of-the-art techniques. Section V concludes this work.

II. Error Tolerance Analysis for Video Coding

In this section, we present an analysis of error tolerance for the HEVC encoder application to derive interesting observations that are leveraged for designing an efficient content-aware approximate computing technique for video coding. Imprecise or approximate computing at the application layer can exploit several basic methods to improve power efficiency by selectively eliminating operations/computations at the algorithm and data levels. In the following, we will first present a brief overview of the HEVC standard followed by our error tolerance analysis.

A. An Overview of the HEVC

The HEVC encoding process is structured as a recursive quad-tree partitioning of the basic Coding Unit (CU - 64x64 pixels). The CU can be divided, in a hierarchical fashion, in four parts from 64x64 to 8x8 as demonstrated in Figure 1. Then, each of these blocks is analyzed by all coding tools in the HEVC software to determine an appropriate compression mode. After that, the best division in terms of coding efficiency (bit rate and visual quality) is chosen. However, the recursive partitioning process does not need to go always up to the deepest 8x8 division. Actually, in many cases, depending upon the CU content properties (for instance, homogeneous blocks with low motion), the 64x64 size is sufficient to have a good coding efficiency. In general, the highly-textured and high-motion regions need to be split into smaller blocks. In contrast, the low-textured and low-motion regions can be encoded using larger blocks. In case, the recursive partitioning stops at the bigger block size (i.e. does not go to the deepest level in the quad-tree), the computational complexity of the coding process can significantly be reduced as all other coding steps will only be applied over the generated partitions. However, the coding efficiency-wise best partition may not exist in the set of generated partitions, which may result in degraded coding efficiency.

For the selected block partitions, coding is done by exploiting the spatial and temporal correlations through so-called *Intra-* and *Inter-Predictions*. While the inter-prediction processes typically employs *matching functions* like sum of absolute differences (SAD), the intra-prediction process employs *prediction generation FIR filters* considering the neighboring pixels. These matching and prediction generation functions are composed of primitive pixel-level operations (like addition, subtraction, and multiplication), which can be approximated at different levels of granularity (ranging from circuit [13], loop perforation [25] to data approximations [19]) while still achieving reasonably good prediction results. For instance, Figure 2 shows different possibilities of applying the *data approximations*, where the ***x:y*** notation means that for a set of **y** pixels only **x** pixels are used in the calculation of the matching and prediction functions. Such data approximations can perturb the algorithm and may lead to different prediction values and modes, and thereby affecting the coding results. Aggressive data approximations may lead to serious video quality degradation.

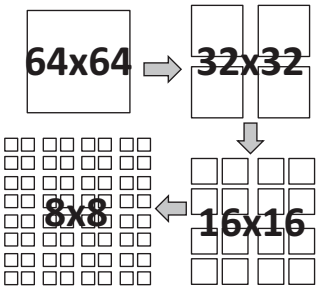


Fig. 1. Quad-tree coding structure.

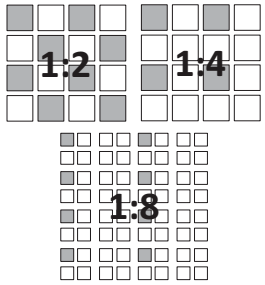


Fig. 2. Data approximations.

B. Analyzing the Error Tolerance of HEVC under Different Application-Level Approximations

Without the loss of generality, we employ two basic application-level approximate computing methods to the HEVC, i.e., (1) *Loop Perforation* [25] to prune the quad-tree coding structure; and (2) *data approximation* through pixel sub-sampling during the prediction process. Using this, we define four different *approximation modes* (*AM-0* to *AM-3*) in order to evaluate the error tolerance of video sequence regions; see Table I. *AM-0* corresponds to no approximation, while *AM-3* corresponds to the most aggressive approximation mode. The approximations at the algorithm level are represented by the maximum quad-tree depth. It limits the depth that the quad-tree coding structure will divide the 64x64 CU. The approximations at the data level are represented by the video data sub-sampling from 1:1 to 1:8. Each *approximation mode* is a composition of a particular algorithm-level approximation and a particular data-level approximation, and it will be applied to every CU in a frame of a given video sequence. Note, in the generic form, an *approximation mode* can be defined as a particular combination of different hardware/software-level approximations from a set of basic approximation methods.

TABLE I. Algorithm and Data Approximation Modes

Approximate Mode	Maximum Quad-Tree Depth	Data Sub-Sampling
AM-0	4 (until 8x8)	1:1
AM-1	3 (until 16x16)	1:2
AM-2	2 (until 32x32)	1:4
AM-3	1 (until 64x64)	1:8

Considering the above-presented *approximation modes*, we have performed a set of experiments to analyze the error tolerance of different recommended test video sequences [26]. A set of six sequences namely *BQMall*, *BasketballDrill* and *RaceHorses* of resolution 832x480, and *BQTerrace*, *BasktetballDrive* and *Cactus* of resolution 1920x1080 were encoded using the HEVC test Model (HM) software version 16.0 [27]. The *approximation modes* are applied CU by CU for all video sequences. From these experiments, final visual quality and computational complexity reduction are obtained. The resulting output visual quality is the metric used to illustrate the error tolerance of a given sequence while computational complexity reduction is used as the abstract metric to show the potential for improving the temperature profile, as temperature is a function of the workload (i.e. computational complexity).

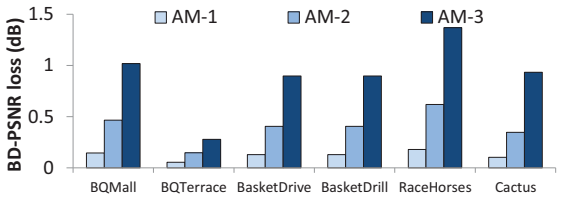


Fig. 3. Quality results for algorithm/data approximate computing modes.

In the first experiment, we have encoded all sequences using all *approximation modes* described in Table I for all CUs in order to check the approximation effects on both visual quality and computational complexity. Figure 3 shows the quality results of this experiment in form of the BD-PSNR loss [28] (in dB) after applying the *approximation modes* AM-1, AM-2, and AM-3 compared to the AM-0 mode (i.e. when no approximations are performed).

It is noticeable that the quality loss increases as the *approximation modes* go further in the quad-tree pruning and in the sub-sampling degrees achieving almost 1.5 dBs of quality loss for the AM-3 in the *RaceHorses* sequence. The algorithm-level approximations in the quad-tree limit the CU division in regions with high motion/texture, where encoding with smaller blocks is essential to get good quality. Also, the data-level approximations imposed by the sub-sampling affects the quality, mainly in heterogeneous regions, where each pixel contributes in a different way to the matching calculation. Moreover, there is a different ratio in the quality loss between sequences, for example, the *BQTerrace* sequence is less affected by the *approximation modes* in comparison with the *RaceHorses* sequence. This happens, since different video sequences have different amount of motion/texture properties and different correlation potential.

Figure 4 shows the impact of using the *approximation modes* on the computational complexity of encoding one frame of the *BasketballDrive* sequence. The workload map is formed by squares that represent the time (normalized to the range of the CU times) to encode each CU in the frame considering all *approximation modes*. Following the quality loss results in Figure 3, the workload reduces as the *approximation modes* prune more the quad-tree structure and the sub-sampling degree increases. The algorithm-level approximations by eliminating some of the quad-tree depths denote the avoidance of the processing of all coding tools used to evaluate each one of the eliminated blocks. Besides, the sub-sampling is applied to the blocks that were not eliminated by the algorithm-level approximations. It is important to note that even with the *approximation modes*, the workload of some CUs do not reduce in the same proportion. In fact, in some cases the CU workload increases as demonstrated in some of the CUs when the AM-3 is applied (Figure 4 (d)). This happens, since

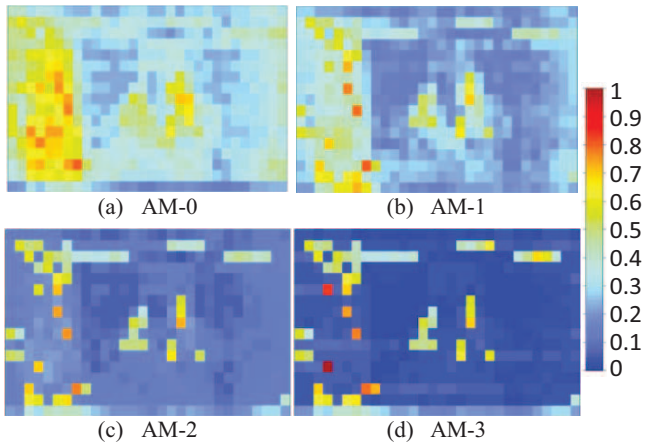


Fig. 4. CUs workload encoding the *BasketballDrive* sequence for all approximation modes.

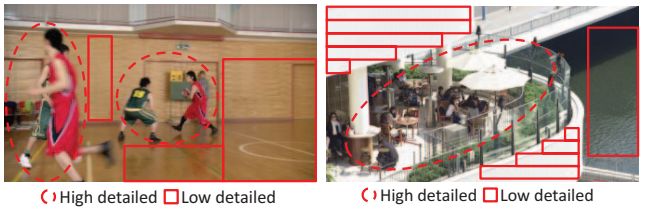


Fig. 5. High/Low detailed regions (1920x1080 pixels).

other coding tools can *negatively* be affected by the quad-tree pruning, and the computational complexity of some algorithm routines may increase (like CABAC) to achieve better coding efficiency.

Although using the proposed *approximation modes* for all CUs seems to be a good way to reduce the workload associated with the video encoding process and consequently to improve the thermal profiles, there are significant quality losses in most of the tested sequences. When the AM-3 mode is applied, there is about 1 dB of BD-PSNR loss for all sequences except the *BQTerrace* and almost 1.5 dB for the *RaceHorses* sequence. When the AM-1 mode is applied the BD-PSNR losses are very low. On the other hand, the total workload reduction may not be sufficient to reduce the temperature of the coding system. Therefore, it is necessary to find a better way to use these *approximation modes* where good quality results can be achieved together with low temperatures.

An interesting observation can be found in the case of *BQTerrace* quality results in Figure 3. The BD-PSNR losses are very low even when the AM-3 is applied. This happens since this sequence have more low texture/motion regions than other sequences, and therefore, using both algorithm-level and data-level approximations do not impact the output quality much. This denotes that there are regions in a video frame that are more resilient to the approximation errors while there are some other regions that are more sensitive to the approximations. Figure 5 shows two frames from two sequences (*BasketballDrive* and *BQTerrace*) where we have identified some low/high detailed areas that can be more or less affected by the approximations explored in this work. The low and high detailed regions are selected based on the texture and motion properties of the objects in a frame.

In order to support the hypothesis that applying the *approximation modes* selectively to some characterized resilient video regions can be less harmful to the final output quality while still contributing to the workload reduction, we have performed a second set of experiments. In these experiments, we selected regions with low texture/motion properties for applying the three *approximation modes* and encoded two frames shown in Figure 5.

Figure 6 shows the BD-PSNR losses where the *approximation modes* being applied to all regions in the frames are compared to applying the *approximation modes* only to those regions characterized as low detailed (i.e., rectangles in Figure 5). Note that the BD-PSNR losses are very low (close to the zero value) for both sequences. Additionally, even when the AM-3 mode is applied, the BD-PSNR losses stay below 0.1 dB, which is almost imperceptible. Our analysis shows that high workload reduction can be achieved at low quality degradation if all three *approximation modes* can be applied selectively to only the resilient regions in the video sequences. We refer to this as *"content-driven adaptive approximate computing"*.

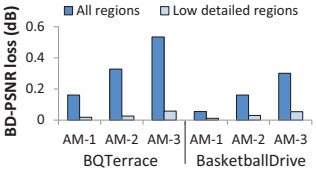


Fig. 6. BD-PSNR losses.

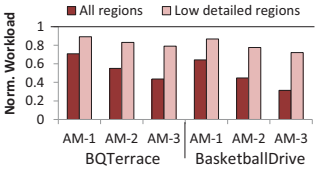


Fig. 7. Workload reduction.

Figure 7 shows the normalized workload of applying the *approximation modes* only in the low-detailed regions and compares it to the workload of applying the *approximation modes* to all regions. It shows that even when the *approximation modes* are applied only to the low-detailed regions, the workload reduction is significant. Considering the AM-3 mode in the low-detailed regions, the *BQTerrace* sequence shows workload reduction close to the AM-1 mode applied in all regions with less than half of the BD-PSNR losses. Moreover, there is a potential of more workload reduction if all the *approximation modes* could be applied to all frame selectively depending upon the properties of their different region.

Summary of Analysis: Approximate computing at the *algorithm-level* by pruning the quad-tree structure and at the *data-level* by sub-sampling the matching function calculation can provide significant workload reductions in the HEVC process. However, in order to improve the tradeoff between workload reduction (and consequently the on-chip temperatures) and the resulted visual quality, different degrees of approximations (i.e. different *approximation modes*) need to be applied adaptively according to the error tolerance/resilience properties of different video regions. The resilience of a video region can be formulated as a function of its texture and motion properties.

III. Thermal Optimization through Adaptive Approximate Computing

In this paper, we present a temperature optimization technique that adaptively employs different *approximation modes* to reduce the temperature associated with the HEVC video coding process. The main goal of our technique is to minimize both temperature and application QoS degradation in terms of BD-PSNR. To achieve such a goal, the temperature optimization at the application-level is obtained through a *content-driven approximate computing technique* that exploits video properties to classify the resilience of different regions of the video sequence in order to control the selection of *approximation modes* during their encoding.

A. Error Resilience Classification

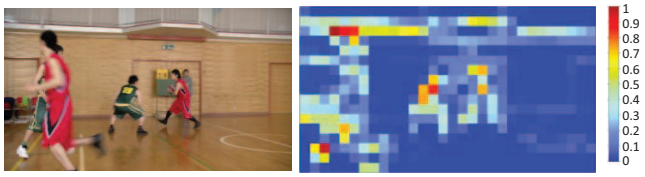
As demonstrated in Section II, some video regions exhibit higher resilience to approximations than others, i.e., tolerate more error and incur low quality loss. It shows that if regions can be classified in terms of their error resilience, it is possible to obtain high workload reductions at low quality penalties. Our analysis in Section II showed that, in general, regions with low motion/texture intensity (low-detailed and homogeneous) are the ones with higher resilience to approximation. On the other hand, regions with high motion/texture intensity (high-detailed and heterogeneous) are more sensitive to approximation errors. Therefore, we formulate the resilience of video regions as a function of their texture and motion properties.

Towards this end, we first estimate the texture and motion intensity of different Coding Units (CUs) of the input video sequence. For each 64x64 CU, we calculate the variance among the luminance samples within the CU (Eq. (1)) using the model of [29]. Then, we compute the normalized variance of all CUs within a frame using Eq. (2), since the most important information from the variance calculations is to compare regions in terms of their error resilience level and identify the ones with the high resilience value.

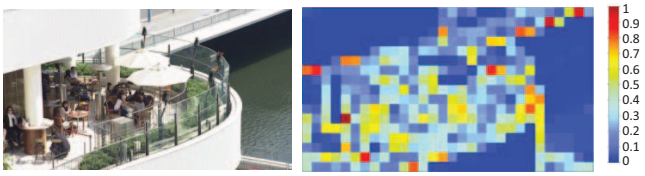
$$v_{CU} = \frac{1}{4096}\sum_{i=1}^{4096}(\rho_i - \rho_{avg})^2 \quad (1)$$

$$norm\_v_{CU} = \frac{v_{CU} - minFrame(v_{CU})}{maxFrame(v_{CU}) - minFrame(v_{CU})} \quad (2)$$

Figure 8 shows a color map where each square represents the normalized variance for each CU. It is possible to observe that the low-variance values indicate low texture/motion regions while high-variance values are found in regions where there is more detailed texture/motion information. Most of the low-variance regions also match with selected regions in the second experiment in Section II.



(a) *BasketballDrive* normalized variance map



(b) *BQTerrace* normalized variance map

Fig. 8. Texture/motion using variance.

Based on this CU-level normalized variance information, we define the following four *resilience levels* to classify each CU using the Eq. (3): *Resilient*, *Medium Resilient*, *Medium Sensitive*, and *Sensitive*. This classification will be used for characterizing different CUs in a video frame and for selecting an *approximation mode* during the CU encoding. Without the loss of generalization and for illustrative purposes, we choose four different resilience levels in order to match with our four *approximation modes* used for the evaluation of the proposed concepts.

$$\Gamma = \begin{cases} Resilient & if(norm\_v_{CU} < Th_{v1}) \\ Medium\ resilient & if(Th_{v1} \le norm\_v_{CU} < Th_{v2}) \\ Medium\ sensitive & if(Th_{v2} \le norm\_v_{CU} < Th_{v3}) \\ Sensitive & if(norm\_v_{CU} \ge Th_{v3}) \end{cases} \quad (3)$$

The threshold values in Eq. (3) were obtained through regression analysis (considering a step size of 0.05) using a small subset of test video sequences with diverse texture/motion properties, which is different from the set of test video sequences used for evaluation, in order to avoid data biasing. For defining the threshold value Th_{v1} corresponding to the level *Resilient*, test video sequences were encoded with the AM-3 mode in the regression analysis. The threshold selection was performed considering the quality degradation behavior and the workload reduction in the regression tests. Similar methodology was adopted for obtaining the other two threshold values Th_{v2} and Th_{v3}. The final thresholds obtained through the regression analysis were 0.1, 0.2 and 0.3 for Th_{v1}, Th_{v2} and Th_{v3}, respectively.

B. Content-Driven Adaptive Approximation Management

Based on the proposed resilience classification of CUs in the HEVC encoding process, our technique uses the obtained threshold values to define the degree of approximations that will be employed for each CU encoding. Algorithm I shows the flow of our light-weight approximate mode selection heuristic. For each frame in a video sequence, the variance of each CU is extracted and added to a variance list *v_list* (line 1 to 4). Then the normalized variance value is calculated for each CU to classify the CU resilience to approximation errors (line 7). According to the classification, one of the four *approx-*

Algorithm I Approximate mode selection heuristic.

```
    Input: sequence S;
1:  v_list = [ ];
2:      for each frame f ∈ S do:
3:      for each CU cu ∈ f do:
4:          v_CU = extract_variance(cu);
5:          update_v_list(v_CU);
6:      for each CU cu ∈ f do:
7:          norm_v_CU = [v_CU – min(v_list)] / [max(v_list) – min(v_list)];
8:          case norm_v_CU:
9:              resilient: encode(cu, AM-3);
10:             medium resilient: encode(cu, AM-2);
11:             medium sensitive: encode(cu, AM-1);
12:             sensitive: encode(cu, AM-0);
```

imation modes is selected in the CU encoding process (line 9 to 12). For CUs that are classified as *Sensitive*, no approximations are performed in the CU encoding.

IV. Results

For evaluation, we perform thermal simulations when encoding different video sequences using HEVC while considering two methodologies. The *first methodology* uses the digital thermal sensor (DTS) [30] from an *Intel i7* processor to measure temperature by the Linux monitoring sensors application. The HM software encoding sequences is restricted to one core and the temperature of this core is read with an interval of 500 milliseconds with only the operational systems basic applications running together with HEVC encoding. The *second methodology* extracts temperature values using an integrated tool chain of GEM5, McPAT, and HotSpot. First, the GEM5 [31] system simulator, using a 32nm 2 GHz out-of-order Alpha as target processor, generates core usage statistics. Then the McPAT [32] power simulator generates power traces using the GEM5 statistics. Finally, the power traces together with the chip layout are used by the HotSpot [33] thermal modeling tool for temperature estimation and for generating the thermal maps. All temperature results for both methodologies were collected using the HEVC test Model (HM) software version 16 and by encoding a set of sequences recommended by the video standardization committee [26]. The variance threshold values were obtained from the *BQTerrace* and *BasketballDrive* sequences. Therefore, to avoid data biasing, we use five additional sequences to evaluate the quality results of our content-driven approximate computing technique and to include a wide range of scenarios considering diverse texture/motion intensity.

Figure 9 shows the thermal profiles extracted using the DTS methodology for encoding four sequences *with* and *without* our approximation technique. It can be observed that the temperature values are smaller when our technique is employed. This happens since there is a significant workload reduction in the encoding process provided by our technique. For the *BasketballDrill* sequence, our technique did not achieve temperature reduction as good as that for the *BasketballDrive*. This can be attributed to the high number of low texture/motion CUs (i.e. more resilient regions) in *BasketballDrive*, and thereby using more aggressive approximations. In contrast, for *BasketballDrill*, due to the more high-textured CUs, less-aggressive or no approximations were employed when encoding those CUs.

Figure 10 shows the average temperature results for all tested sequences using the DTS methodology. Our content-driven adaptive approximate computing technique achieved low temperatures for all sequences in comparison to the regular HEVC encoding without any approximations. It can be observed that our technique significantly improves the thermal profiles (e.g., 15 ºC reduced temperature in *Cactus*), though there are some sequences where the average temperature reduction is only 5 ºC (i.e. *BQMall*). On average, for all sequences, our technique achieved 10 ºC of temperature reduction. The difference in temperature reduction across different sequences is attributed to their diverse distributions of low-resilience CUs that limit the applicability of aggressive approximations.

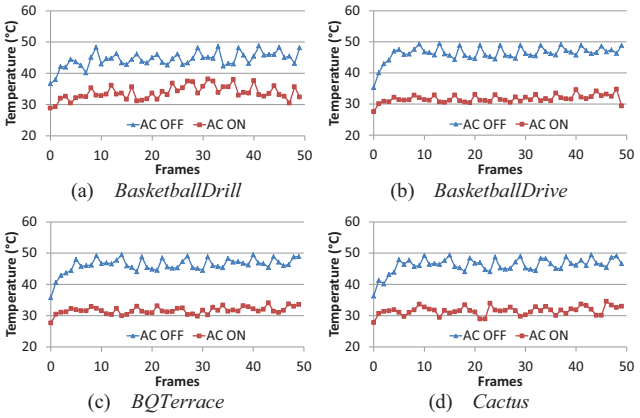


Fig. 9. Temperature profile for different frames.

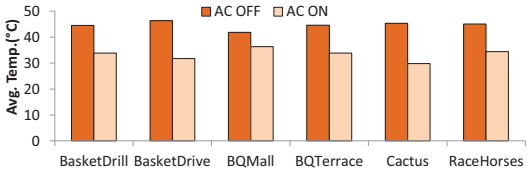


Fig. 10. Average temperature for all tested sequences.

Figures 11 and 12 show the thermal maps extracted from the second evaluation methodology that uses the HotSpot thermal modeling tool. We show thermal maps only for two sequences due to space limitation. From these figures it can be observed that our technique successfully improves the temperature profile of video coding systems. The left-side hotter maps represent the steady-state temperature when using the regular HEVC encoding *without* any approximations, while the right-side colder maps represent the steady-state temperature of the HEVC encoding when *using* our technique. In summary, the temperature reduction for all sequences using the HotSpot methodology was also about 10 ºC on average.

Figure 13 shows the impact of our technique on the output quality loss (in terms of BD-PSNR loss) when compared to the regular HEVC encoding *without* using our technique. The quality degradation for all tested cases is on average 0.48 dBs (excluding sequences that were used on training the threshold values). Even for the worst-case results (i.e. the *RaceHorses* sequence), our technique incurs a maximum BD-PSNR loss of 0.66 dBs, while in the best-case (i.e., the *Keiba* sequence) it incurs the maximum BD-PSNR loss of only 0.31 dBs. We also compared the BD-PSNR losses of our technique with statically using the AM-3 mode, i.e., most aggressive approximation mode with the highest savings. For all cases our technique presents better output quality results than employing only the AM-3, which is attributed to the resilience-driven adaptive approximation control.

We compare the quality results of our approximate computing-based temperature optimization technique with the dynamic thermal management policy proposed in [21][22] that use frame drops rates varying from 5.7% to 83.3% to keep the temperature under a defined threshold. For fairness of comparison, we consider that a drop frame means that the encoder replaces it by the information of the last-encoded frame and skips the encoding process. We also compare quality results with the dynamic thermal management work [5] that uses run-time encoder configuration to keep the temperature under a

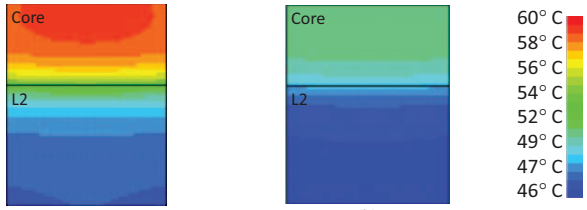


Fig. 11. Thermal maps of *BasketballDrill* (a) AC OFF (b) AC ON

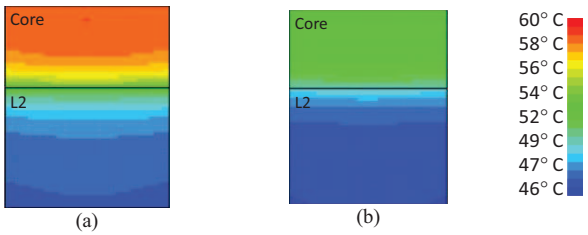


Fig. 12. Thermal maps of *BQMall* (a) AC OFF (b) AC ON

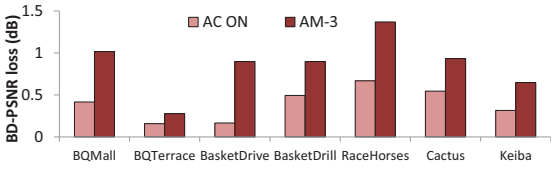


Fig. 13. Quality results.

defined threshold. The quality results are reported corresponding to the same temperature reduction cases, e.g., 10 ºC reduction.

Figure 14 shows the quality comparison in terms of PSNR for encoding four sequences with the HEVC. Our technique is able to outperform the state-of-the-art techniques in most of the evaluated cases. When comparing with the frame-drop based technique, our approximate computing based technique presents the best results in all cases. The quality degradation of dropping 20% of the frames leads to a PSNR loss of 8 to 16 dBs. This significant quality loss is usually not tolerable for users on encoding devices. When comparing with the thermal management policy that change encoder configurations at run-time, our technique presents better results for three sequences and similar result for the *BQMall* sequence. Encoder configuration, mainly the quantization parameter (QP), can significantly degrade quality. Higher QP values introduce error in the encoding process, generating quality loss. When applied to sequence regions that are too sensitive to errors, it may increase the quality losses. On the other hand, our technique takes into account the texture and motion properties of different sequence regions and adaptively employs varying degree of approximations. Consideration of resilience properties of different regions help in achieving better quality results than other techniques that do not account for such information.

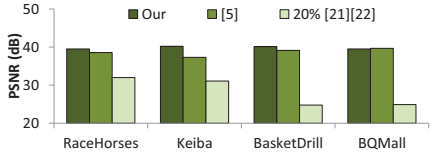


Fig. 14. Comparison of quality results with state-of-the-art works.

V. Conclusions

In this paper, we have introduced a temperature optimization technique using adaptive approximate computing for video coding systems. It employs an adaptive content-driven approximate computing technique that classifies regions of a given video sequence w.r.t. their resilience properties as a function of their texture and motion characteristics. Based on the resilience level, different approximation modes are employed to reduce the on-chip temperature of the High Efficiency Video Coding process. Different approximation modes are realized through both algorithm-level and data-level approximation computing mechanisms. Our results illustrate the temperature reduction potential of our adaptive approximate computing technique and benefit of considering the resilience knowledge to lower the output video quality losses compared to several state-of-the-art techniques. This work shows that approximate computing bears a significant potential for temperature optimization when employed selectively considering the variable resilience properties of the input data sets.

Acknowledgment

This work is supported in parts by the German Research Foundation (DFG) as part of the priority program Dependable Embedded Systems (SPP 1500 – http://spp1500.itec.kit.edu).

References

[1] Cisco, "Cisco Visual Networking Index: Forecast and Methodology, 2012, 2017", May 2013.

[2] ITU-T, "SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS – Infrastructure of Audiovisual Services – Coding of Moving Video – High Efficiency Video Coding." Apr. 2013.

[3] J. Vanne, M. Viitanen, T. Hämäläinen, A. Hallapuro, "Comparative Rate-Distortion-Complexity Analysis of HEVC and AVC Video Codecs", *IEEE TCSVT*, v. 22, issue 12, pp. 1885 – 1898, 2012.

[4] D. Palomino, M. Shafique, H. Amrouch, A. Susin, J. Henkel, "hevcDTM: Application-Driven Dynamic Thermal Management for High Efficiency Video Coding", DATE, 2014.

[5] D. Palomino, M. Shafique, A. Susin, J. Henkel. "TONE: Adaptive Temperature Optimization for the Next Generation Video Encoders". ISLPED, pp. 33-38, 2014.

[6] R.H. Dennard et. al. "Design of ion-implanted MOSFETs with very small physical dimensions". *IEEE Journal of Solid-State Circuits*, 256-268, 1974.

[7] M. Bohr. "A 30 year retrospective on Dennard's MOSFET scaling paper". *IEEE Solid-State Circuits Society Newsletter*, 12(1):11-13, Winter 2007.

[8] M. Shafique, S. Garg, J. Henkel, D. Marculescu, "The EDA challenges in the dark silicon era", DAC, 2014.

[9] J. Henkel, L. Bauer, N. Dutt, P. Gupta, S. Nassif, M. Shafique, M. Tahoori, N. Wehn, "Relieble on-chip systems in the nano-era: Lessons lerant and furture trends", DAC, 2013.

[10] D. Gnad, M. Shafique, F. Kriebel, S. Rehman, D. Sun, J. Henkel, "Hayat: Harnessing Dark Silicon and Variability for Aging", DAC, 2015.

[11] S. Venkataramani, S. Chakradhar, K. Roy, A. Raghnathan. "Computing Approximately, and Efficiently". DATE. pp. 748-751, 2015.

[12] S. Ramasubramanian, S. Venkataramani, A. Parandhaman, A. Raghunathan, "Relax-and-retime: A methodology for energy-efficient recovery based design. DAC, 2013.

[13] V. Gupta, D. Mohapatra, S. Park , A. Raghunathan, K. Roy, "IMPACT: imprecise adders for low-power approximate computing. ISLPED, 2011.

[14] V. Chippa, D. Mohapatra, A. Raghunathan, K. Roy, S. Chakradhar, "Scalable effort hardware design", *IEEE Trans. On VLSI Systems*, 2014.

[15] V. Chippa, S. Venkataramani, K. Roy, A. Raghunathan, "StoRM: a stochastic recognition and mining processor", ISLPED, 2014.

[16] S. Venkataramani, et. al. "AxNN: energy-efficient neuromorphic systems using approximate computing", ISLPED, 2014.

[17] S. Chakradhar, A. Raghunathan, "Best-effort computing: Re-thinking parallel software and hardware" DAC, 2010.

[18] G. Correa, P. Assuncao, L. Agostini, L. Cruz. "Fast HEVC Encoding Decisions Using Data Mining". *IEEE TCSVT*, pp. 660-673, 2015.

[19] M. Shafique, B. Zatt, F. Walter, S. Bampi, J. Henkel, "Adaptive power management of on-chip video memory for Multiview video coding," DAC, pp. 866-875, 2012.

[20] H. Khdr, T. Ebi, M. Shafique, A. Amrouch, J. Henkel, "mDTM: Multi-objective dynamic thermal management for on-chip systems", DATE, 2014.

[21] W. Lee, K. Patel, M. Pedram, "Dynamic thermal management for MPEG-2 decoding," ISLPED, pp. 316-321, 2006.

[22] W. Lee, K. Patel, M. Pedram, "GOP-Level dynamic thermal management in MPEG-2 decoding", TVLSI, vol.16, no.6, pp. 662-272, 2008.

[23] J. Osterman, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, T. Wedi, "Video coding with H.264/AVC: tools, performance, and complexity", IEEE CS Magazine, vol. 4, no. 1, 2004.

[24] A. Mirtar, S. Dey, A. Ragunathan, "Adaptiation of vídeo encoding to address dynamics thermal management effects", IGCC, pp. 1-10, 2012.

[25] S. Douskos, S. Misailovic, H. Hoffman, M. Rinard, "Managing performance vs. accuracy trade-offs with loop perforation", ESEC/FSE, 2011.

[26] F. Bossen, "Common test conditions and software reference configurations, ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC) document JCTVC-K1100, October 2012." Tech. Rep. 2012.

[27] Joint Collaborative Team on Video coding (JCT-VC), "HM 16.0 Reference Software" [Online]. Available at: http://hevc.hhi.fraunhofer.de/

[28] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves, Tech. Rep. VCEG-M33, ITU-T Video Coding Experts Group (VCEG), 2001." Tech. Rep., 2001.

[29] M. Shafique, B. Zatt, and J. Henkel, "A complexity reduction scheme with adaptive search direction and mode elimination for multiview video coding," in Picture Coding Symposium (PCS), pp. 105–108, 2012.

[30] M. Berktold, T. Tian, "CPU monitoring with DTS/PECI", Sep. 2010.

[31] N. Binkert, et. al. "The gem5 simulator," ACM SIGARCH Computer Architecture News, vol. 39, no. 2, pp. 1–7, 2011.

[32] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures," MICRO-42. 2009.

[33] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-aware microarchitecture," in ACM SIGARCH Computer Architecture News, vol. 31, pp. 2–13, 2003.