UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Programa de Pós-Graduação em Computação

Dissertação

Energy/Quality-Aware Hardware Solutions for the Residual Coding Loop Components of the High Efficiency Video Coding Standard

Luciano Almeida Braatz

Pelotas, 2018

Luciano Almeida Braatz

Energy/Quality-Aware Hardware Solutions for the Residual Coding Loop Components of the High Efficiency Video Coding Standard

Dissertação apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Mestre em Ciência da Computação

Advisor: Prof. Dr. Marcelo Schiavon Porto
Coadvisors: Prof. Dr. Daniel Munari Palomino
Prof. Dr. Luciano Volcan Agostini

Pelotas, 2018

Universidade Federal de Pelotas / Sistema de Bibliotecas
Catalogação na Publicação

B794e Braatz, Luciano Almeida

Energy/quality-aware hardware solutions for the residual coding loop components of the high efficiency video coding standard / Luciano Almeida Braatz ; Marcelo Schiavon Porto, orientador ; Daniel Munari Palomino, Luciano Volcan Agostini, coorientadores. — Pelotas, 2018.

111 f.

Dissertação (Mestrado) — Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 2018.

1. Video coding. 2. HEVC. 3. Residual coding loop. 4. Hardware design. 5. Energy/quality awareness. I. Porto, Marcelo Schiavon, orient. II. Palomino, Daniel Munari, coorient. III. Agostini, Luciano Volcan, coorient. IV. Título.

CDD : 005

Elaborada por Maria Inez Figueiredo Figas Machado CRB: 10/1612

AGRADECIMENTOS

Aos meus pais, Nelson e Mariza, pelo apoio e por sempre acreditarem no meu sucesso.

Aos familiares e amigos, pela compreensão da ausência durante esta jornada.

Aos professores Marcelo Porto, Daniel Palomino, Luciano Agostini e Bruno Zatt pelo auxílio na idealização do tema, sua orientação, paciência e apoio.

Aos colegas do Grupo de Arquiteturas e Circuitos Integrados pela parceria e ajuda no desenvolvimento deste projeto.

Theorie ohne Praktik ist verwirrt;
Praktik ohne Theorie ist unzulänglich.
— G. J. VOGLER

ABSTRACT

BRAATZ, Luciano Almeida. **Energy/Quality-Aware Hardware Solutions for the Residual Coding Loop Components of the High Efficiency Video Coding Standard**. 2018. 111 f. Dissertação (Mestrado em Ciência da Computação) – Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2018.

Multimedia applications, such as digital videos, are very popular nowadays, especially on mobile devices. Moreover, there is an expectation of continuous growth of the Internet-based digital videos traffic throughout the next years. Video coders, e.g. the High Efficiency Video Coding (HEVC), are very important in this context as the video coders can rationalize Internet resources by reducing the amount of video-related data flowing through the network. Unfortunately, this data reduction requires a huge computational effort. Thus, hardware accelerators can be used as a feasible solution, providing high-throughput on low-power. The HEVC residual coding loop (RCL), composed of direct transform, direct quantization, inverse quantization, and inverse transform, is a highly-requested stage of video coding standards since it is used multiple times to test several coding modes (CMs). Therefore, the objective of this work is to provide multiple energy/quality-aware dedicated hardware solutions to increase throughput for the components of RCL in HEVC, allowing real time processing of many CM by the RCL. Thus, the HEVC encoder using the presented solutions is expected to be more coding efficient than the ones using the solution proposed by the existing related works. Innovative solutions are proposed in this work to increase throughput, which has direct impact on the coding efficiency in the RCL components, with low power dissipation. A direct DCT was proposed based on the Fast Fourier Transform (FFT), which allows an intensive hardware reuse, and energy-consumption reduction, able to operate up to 2.54 GHz while dissipating 12.33 mW of power; an energy/quality scalable inverse DCT is presented using a bypass engine based on statistical analysis, setting a trade-off between coding-efficiency and energy-efficiency, which operates up to 737.46 MHz and dissipates between 13.87 mW and 16.84 mW; and a project space exploration of quantization architectures is also presented, with power dissipation of 152.13 mW and 31.15 mW at 888.10 Mhz and 1.36 GHz, respectively, including an integrated direct/inverse quantization, which reduces the number of arithmetical operations by integrating direct quantization and inverse quantization, dissipating 369.37 mW at 1.68 GHz. The developed hardware architectures are able to process up to 108, 32, 38, 58, and 72 CM of UHD 4K videos at 60fps for the DCT, IDCT, direct quantization, inverse quantization, and integrated direct/inverse quantization hardware architectures, respectively. When compared with

related works, the developed RCL components hardware architectures can operate on higher frequencies, present higher throughputs, and are more energy efficient.

Keywords: video coding; hevc; residual coding loop; hardware design; energy/quality awareness

RESUMO

BRAATZ, Luciano Almeida. **Soluções em Hardware com Foco em Aumento da Qualidade e Economia de Energia para os Componentes do Laço de Codificação Residual do Padrão HEVC** . 2018. 111 f. Dissertação (Mestrado em Ciência da Computação) – Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2018.

Aplicações multimídia, como vídeos digitais, são muito populares hoje em dia, especialmente em dispositivos móveis. Além disso, há uma expectativa de crescimento contínuo do tráfego de vídeos digitais baseados na Internet nos próximos anos. Codificadores de vídeo, como o HEVC (do inglês *High-Efficiency Video Coding*), são muito importantes neste contexto, pois os codificadores de vídeo podem racionalizar os recursos da Internet reduzindo o tráfego de dados relacionados a vídeos na rede. Infelizmente, essa redução no volume de dados requer um enorme esforço computacional. Assim, o uso de aceleradores de hardware se apresenta como uma solução viável, pela sua capacidade de apresentar alto *throughput* e baixo consumo energético. O laço de codificação residual do HEVC (RCL, do inglês *Residual Coding Loop*), composto por transformada direta, quantificação direta, quantização inversa e transformada inversa, é um estágio altamente solicitado de padrões de codificação de vídeo, pois é usado para testar vários modos de codificação. Portanto, o objetivo deste trabalho é fornecer soluções de hardware dedicadas com foco em eficiência energética e no aumento do *throughput* dos componentes do RCL no HEVC, permitindo que os componentes do RCL possam processar diversos modos de codificação em tempo real. Assim, o codificador HEVC que usa as soluções apresentadas deverá ser mais eficiente em termos de codificação do que os que usam as soluções propostas pelos trabalhos relacionados existentes. Este trabalho propõe soluções inovadoras para aumentar o *throughput* dos componentes do RCL, com impacto direto na eficiência de codificação e baixa dissipação de energia. A DCT, proposta com base na Transformada Rápida de Fourier (FFT, do inglês *Fast Fourier Transform*), permite a reutilização intensiva de hardware e redução de consumo de energia, sendo capaz de operar a até 2,54 GHz enquanto dissipa 12,33 mW de potência; uma IDCT com compromisso entre energia e qualidade é apresentada usando um mecanismo de *bypass* com base em análise estatística, estabelecendo um compromisso entre eficiência de codificação e eficiência energética, opera até 737,46 MHz e dissipa entre 13,87 mW e 16,84 mW; uma exploração do espaço de projeto de arquiteturas de quantização também é apresentada, com uma dissipação de potência entre 152,13 mW e 31,15 mW a 888,10 Mhz e 1,36 GHz, respectivamente; um módulo integrado de quantização direta e in-

versa que reduz o número de operações aritméticas integrando quantização direta e quantificação inversa, dissipando 369,37 mW a 1,68 GHz. As arquiteturas de hardware desenvolvidas são capazes de processar até 108, 32, 38, 58 e 72 modos de codificação de vídeos UHD 4K a 60fps para DCT, IDCT, quantização direta, quantização inversa e a arquitetura integrada de quantização direta e inversa, respectivamente. Comparado aos trabalhos relacionados, as arquiteturas de hardware dedicadas para os componentes da RCL operam com frequências mais altas, apresentam throughputs maiores e são mais eficientes energeticamente.

Palavras-Chave: codificação de vídeo; hevc; laço de codificação residual; projeto de hardware; foco em qualidade e energia

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

1D-DCT	One-dimensional Discrete Cosine Transform
1D-IDCT	One-dimensional Inverse Discrete Cosine Transform
2D-DCT	Two-dimensional Discrete Cosine Transform
AZC	Almost Zeroed Column
BCP	Bypassed Columns Percentage
BC	Butterfly Cluster
BD-Rate	Bjøntegäard Distance on Bit Rate
BD-BR	BD-Rate
BRO	Bit-Reversal Ordering
CB	Coding Block
CM	Coding Mode
#CM	Number of Coding Modes
CPU	Central Processing Unit
CTB	Coding Tree Block
CTC	Common Test Conditions
CTU	Coding Tree Unit
CU	Coding Unit
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DIT	Decimation-in-time
DQ-SSC	Direct Quantization Single-Sample Core
DSP	Digital Signal Processor
DST	Discrete Sine Transform
EOD	Even/Odd Decomposition
EOS	Even/Odd Separation
eps	Energy per Sample

EQS	Energy/Quality Scalable
FDQ	Frequency-Dependent Quantization
FFT	Fast Fourier Transform
FPGA	Field Programmable Gate Array
fps	Frames per Second
FWVGA	Full Wide Video Graphic Array
GM	Generic Multiplication
GPU	Graphical Processing Unit
HD	High Definition
HEVC	High Efficiency Video Codign
HM	HEVC Test Model
HVS	Human Visual System
IDIQ	Integrated Direct/Inverse Quantization
IEC	International Electrotechnical Commission
IQ-SSC	Inverse Quantization Single-Sample Core
ISO	International Standardization Organization
ITU-T	International Telegraph Union, Telecommunication Standardization Sector
JCT-VC	Joint Collaborative Team on Video Coding
LCR	Laço de Codificação Residual
LUT	Look-Up Table
MCM	Multiplierless Constant Multiplication
MPEG	Motion Picture Experts Group
MPSoC	Multi-Processed System On Chip
N/A	Not applicable
N/D	Not defined
OU	Operational Unit
PB	Prediction Block
PIDIQ	Parallel Integrated Direct/Inverse Quantization
PU	Prediction Unit
PVT	Process Voltage-Temperature
QM	Quantization Matrix
QP	Quantization Parameter
QStep	Quantization Step
QTR	Quantized Transformed Residual

R-D	Rate-Distortion
RCL	Residual Coding Loop
RC	Cadence Encounter RTL Compiler
RDO	Rate-Distortion Optimization
RD	Residual Data
ROM	Read-Only Memory
RQT	Residual Quadtree
RR	Reconstructed Residuals
RTL	Register-Transfer Level
spc	Samples per Cycle
sps	Samples per Second
SSC	Single-Sample Core
TB	Transform Block
TCF	Toggle Count File
TR	Transformed Residual
TU	Transform Unit
UHD	Ultra-High Definition
USB	Universal Serial Bus
VCD	Value Change Dump
VCEG	Video Coding Experts Group
VQVGA	Wide Quarter Video Graphic Array
WQXGA	Wide Quad Extended Graphic Array

CONTENTS

1 INTRODUCTION

Multimedia applications, such as digital videos, are very popular nowadays, especially on mobile devices. According to CISCO (2017), 82% of total Internet traffic in 2021 will be related to digital video content. For instance, it would take more than 5 million years for a person to watch the amount of video content crossing internet monthly in 2021. CISCO (2017) also predicted a 19.668 PBytes (petabytes) monthly traffic of digital video content in mobile devices in 2021. Such traffic is more than 11 times bigger than the monthly traffic registered in 2016 (1.756 PBytes).

Nonetheless, uncompressed digital videos require a prohibitive amount of data for their representation. For instance, an UHD 4K video, with a resolution of 3840x2160 pixels, using 24 bits per pixel, and with a frame rate of 120 frames per second (fps), requires 23.89Gbps to broadcast a 10-minute video, or 1.79TBytes for its storage.

Fortunately, digital video data present high level of redundancy. Therefore, much of the information used to represent digital videos can be suppressed by referencing previously processed digital video data. Video coding solutions, such as the state-of-the-art in video coding - High Efficiency Video Coding (HEVC) (ITU-T, 2015), exploit such characteristics. Video coders explore techniques which offer new ways to represent digital videos, suppressing redundant data. Thus, it is possible to encode a digital video reducing the amount of data required to representing the digital video in up to two orders of magnitude (AGOSTINI, 2007). On the other hand, the techniques used in video coders can demand a huge computational effort, due to the amount of data to be processed, and how thoroughly the encoding algorithms search for redundancies (CORREA et al., 2012).

Therefore, encoding ultra-high definition videos in real time using software solutions running on general-purpose processors is difficult to achieve, due to the above-mentioned computational effort requirements. Nonetheless, software-based video encoders are able to encode high-resolution videos in real time by requiring high parallelism level (multithreading), processors operating in high-frequency, and/or significant simplification of the video encoding algorithms. Hence, software-based real-time video encoders have expensive, energy-consuming hardware requirements.

Devices with digital video capabilities, such as mobile devices have limited resources, such as CPU, memory, and energy, i.e. the hardware configurations of mobile devices hardly comply with the hardware requirements of software-based real-time video encoders. Therefore, it is important to develop high-throughput hardware architectures implementing video coding tools to reduce coding time and to allow real-time encoding of beyond high-resolution videos.

Mobile devices can benefit from high-throughput video-encoding dedicated hardware architectures since their use allow the mobile devices to encode digital video in real-time. The real-time digital video encoding capability is important to avoid the use of storage space in the time span between capturing and encoding digital videos, as well as improving the user experience since the user can watch a video as soon as he finishes recording it. The use of video-encoding dedicated hardware architectures supports mobile devices energy-saving requirements since such hardware architectures can be powered off while they are not in use or the energy consumption can be dosed according to the current video encoding quality.

The residual coding loop (RCL), composed of direct transform, direct quantization, inverse quantization, and inverse transform, is a highly-requested stage of the video coding standards since it is used multiple times to test several coding modes (CM) in the prediction stage (BUDAGAVI; FULDSETH; BJONTEGAARD, 2014). Based on the results of the CM testing, the encoder chooses the best trade-off between the encoded video file size and the visual quality of the decoded video. Experiments with the HEVC Test Model, version 16.7 (HM16.7), the reference software of the HEVC, in chapter 3 showed RCL processing between 36 and 267 times the number of samples in the video data, on average, while testing different CM. Given this processing requirements, encoding UHD videos in real time requires high-frequency operated hardware architectures, e.g. RCL components hardware architectures must operate at $2.0GHz$ to be able to encode UHD 4K videos at 60fps following the same behavior of the HM16.7. RCL components hardware architectures operating at lower frequencies reduce the capability of processing as many CM as the hardware architecture operating under the specified full-frequency. Hence, the reduction of the working frequency of RCL components hardware architectures lead to a bottlenecked processing capability of the encoder. Thus, less CM can be tested, which can lead to suboptimal CM choices and consequent losses in the encoder coding efficiency. Therefore, the RCL must present the highest throughput, the lowest latency, and the lowest possible power dissipation. The throughput increase allows increasing the number of tested CM. Thee low latency can improve the overall timewise performance of the encoder, since the intermediary RCL results can lead to early termination of the encoding process. And, the low power dissipation is essential while being used in battery-powered devices.

There are several works in the literature focusing on the different components of

the RCL, such as BONATTO et al. (2017), MASERA; MARTINA; MASERA (2017), RENDA et al. (2017), BASIRI; MAHAMMAD (2017); CHATTERJEE; SARAWADEKAR (2017), ANSARI; MANSOURI; AHAITOUF (2016), JRIDI; MEHER (2016), and DIAS; ROMA; SOUSA (2015). The architecture proposed in those works support processing between one (ANSARI; MANSOURI; AHAITOUF, 2016) and 245 (BASIRI; MAHAMMAD, 2017) CM in HD 1080p videos at 30fps, and between one (BONATTO et al., 2017) (CHATTERJEE; SARAWADEKAR, 2017) and 30 (BASIRI; MAHAMMAD, 2017) CM in UHD 4K videos at 60fps. However, experiments in chapter 3 show that optimal encoding can be achieved by processing up to 267 CM. Moreover, the processing restriction in the related works requires an encoder algorithmic simplification which can decrease the encoder efficiency. In conclusion, the related works seem to overlook their coding losses due to their usual single CM processing targets. The usage of hardware architectures for RCL components with higher throughput capability allows the exploration of multiple CM in real time. As the number of CM processed increases, it is more likely to find the best CM (global maxima, instead of the local maxima provided by related works). Therefore, the limited throughput in related works can have a significant (and negative) impact on the encoder coding efficiency, regarding the trade-off between the quality and the size of the encoded video. In other words, the claim of real-time encoding in related works should be bound by the coding efficiency loss due to the limitation of the number of CM processed, since RCL components hardware designs must be able to process up to the above mentioned 267 CM to be called real-time RCL components hardware designs.

The objective of this work is to develop multiple energy/quality-aware dedicated hardware solutions to increase the throughput of the components of RCL in HEVC standard, which allows processing dozens of CM in high-resolution videos in real time. In such context, the contributions of this work are:

- An analysis of the throughput requirements of the HEVC RCL components;
- An analysis of the impact of the throughput limitation on the encoded video size;
- A frequency-dependent quantization tool impact analysis on the encoded video (size and quality) and on the hardware architecture (area, frequency and power);
- A dedicated hardware design of an FFT-inspired direct 1D-DCT;
- A regular 1D-IDCT dedicated hardware design and energy/quality scalable 1D-IDCT dedicated hardware design;
- A project space exploration of the direct and inverse quantization hardware architectures;
- An integrated direct/inverse quantization hardware architecture.

This work proposes innovative dedicated hardware solutionsfor the HEVC RCL components, focusing in low power, higher throughput and better coding efficiency. The 1D-DCT presented is based on the Fast Fourier Transform (FFT), which allows an intensive hardware reuse and consequent energy-consumption reduction. Furthermore, an energy/quality scalable 1D-IDCT is presented using a bypass engine based on statistical analysis, allowing the adjustment of the trade-off between coding-efficiency and energy-efficiency. Moreover, a project space exploration of quantization architecture is made to maximize throughput- and energy-efficiency of the direct- and inverse-quantization, considering the flat-quantization and the frequency dependent quantization (FDQ), the two quantization methods supported by HEVC. Also, an integrated direct/inverse quantization is presented, which improves the throughput and reduces energy consumption, by reducing the number of arithmetical operations through the integration of the direct quantization and the inverse quantization.

The remaining of this work is organized as follows: chapter 2 describes the HEVC encoder, its resources and the more relevant encoding tools for this master thesis, chapter 3 presents statistical analyses regarding the RCL use, impact of limiting RCL usage, and the cost-effectiveness of using an optional HEVC coding tool, chapter 4 describes the RCL components hardware solutions designed during this work, chapter 5 presents the results of the hardware solutions described in chapter 4 and compares them with related works, and chapter 6 presents the conclusions and future works.

2 HIGH EFFICIENCY VIDEO CODING

The High Efficiency Video Coding (HEVC) standard, developed by the Joint Collaborative Team on Video Coding (JCT-VC), is considered the state-of-the-art in video coders. The JCT-VC is composed of members of the Video Coding Experts Group (VCEG), from the International Telegraph Union, Telecommunications Standardization Sector (ITU-T), and members of the Motion Picture Experts Group (MPEG), from the International Standardization Organization/International Electrotechnical Commission (ISO/IEC) technical committee. HEVC was released in 2013 under the premise of reducing by half the video size when compared with the previous encoding video standard, the H.264 (ITU-T, 2003) , while maintaining the encoded video objective quality (ITU-T, 2015). The encoding efficiency of HEVC is due to the adoption of several new or improved coding tools. The following sections describe the operation of the HEVC, and detail the resources available for HEVC – the HEVC Test Model (HM) reference software (FLYNN et al., 2011) and the Common Test Conditions (CTCs) (BOSSEN, 2013) – to validate and compare results of works related to HEVC.

2.1 HEVC Operation

The HEVC standard uses block-based hybrid coding techniques (SCHWARZ; SCHIERL; MARPE, 2014). Therefore, digital videos are encoded through video samples prediction, prediction errors (residuals) transformation, and entropy encoding of prediction information and transformed residuals. The video samples used in the prediction stage are sample blocks extracted from video frames. Figure 1 presents the data flow for HEVC encoder.

The original frame is the raw digital video frame input, which is partitioned into small blocks by the partitioning stage to be processed by the following stages. In the prediction stage, the current block is compared with the reference frames, in a search for the most similar block. The prediction stage can use one of two prediction modes:

- *Intra-picture prediction*: In the intra-picture prediction, the decoded border of adjacent blocks are used as reference data for spatial prediction. Intra-picture predic-

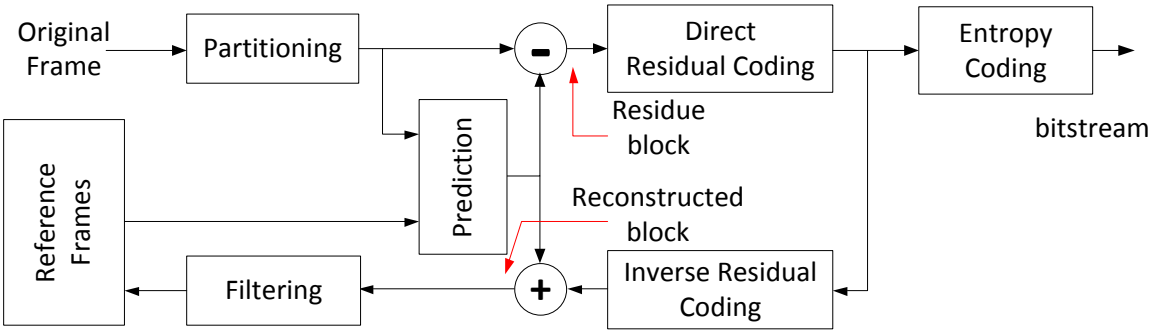

Figure 1 – HEVC encoder data flow

tion supports 33 directional modes, plus planar prediction mode (surface fitting), and DC (flat) prediction mode. The selected intra-picture prediction modes are encoded by deriving most probable modes based on modes used in previously encoded PBs (LEINEMA; HAN, 2014).

- *Inter-pictures prediction*: The encoding process for inter-pictures prediction consists of choosing motion data comprising the selected reference picture and motion vector to be applied for the prediction of the samples of each block. The encoder and decoder generate identical inter-pictures prediction by applying motion compensation using the motion vector and mode decision data (SULLIVAN et al., 2012).

The most similar block is subtracted from the original block to generate the residue block, which will be processed by the direct residual coding stage and inverse residual coding stage. The direct residual is composed of the direct transform component and the direct quantization component. The direct transform component transforms residual data (RD) from the spatial domain to transformed residuals (TR) in the frequency domain. The direct transform concentrates the information energy in few low-frequency coefficients in TR. The direct quantization scales down TR, reducing the amplitude for low-frequency TR and setting most high-frequency TRs to zero, which improves the compression efficiency of entropy coding. The quantized transformed residues (QTR) are used by the entropy coding to calculate the bit-rate used in the Rate-Distortion Optimization (RDO) tool or to generate the output bitstream of the coded video. The QTR data are also processed by the inverse quantization and, subsequently, by the inverse transform. The reconstructed residues (RR) are combined with block prediction data to reconstruct the block as it would be reconstructed in the decoder, to generate the distortion metric used by RDO, or to be used as reference in next blocks predictions. Therefore, the RCL components have high-processing demand. Thus, hardware-implemented RCL components are required to achieve real-time encoding of high-resolution videos.

The RDO calculates the Rate-Distortion cost (RDCost) for each CM and chooses the CM with the lowest RDCost for the actual encoding. Bit-rate and distortion metrics are combined into the RDCost, expressed in (1), where D is the distortion metric, R is the bit-rate metric, and λ is the Lagrange multiplier, which balances the trade-off between distortion (quality) and bit-rate (size) (SCHWARZ; SCHIERL; MARPE, 2014).

$$RD_{Cost} = D + \lambda \times R \tag{1}$$

The focus of this work is the high-throughput, low-power hardware implementations of RCL components (direct residual coding and inverse residual coding, in Figure 1. Therefore, the RCL and its components will be detailed in section 2.1.2. Nonetheless, the HEVC partitioning structure (partitioning stage, on Figure 1) must be further detailed, in section 2.1.1, to better understand the RCL operation.

2.1.1 Partitioning Structure

In HEVC partition structure there are four types of units, with distinct characteristics and functions: Coding Tree Units (CTU), Coding Units (CU), Prediction Units (PU), and Transform Units (TU) (SULLIVAN et al., 2012). Furthermore, there are four block types: Coding Tree Blocks (CTB), Coding Blocks (CB), Prediction Blocks (PB), and Transform Blocks (TB). The blocks are bi-dimensional collections of data associated to one of the color channels in the samples of a unit. For instance, a PU is composed of one luminance PB (Y channel) and two chrominance PBs (U and V channels). Therefore, there is an equivalent relationship between TUs and TBs, between CUs and CBs, and between CTUs and CTBs. On the other hand, the units follow a hierarchy such as a CTU is a collection of CUs and a CU is a collection of PUs and TUs. The division occurs in a structure called quadruple-tree (quadtree).

The quadtree takes the highest-hierarchy unit as its root node. The root node can remain whole or be split into four nodes of the same size. Each resulting node can be further split or not, recursively, as long as the node size is bigger than the minimum node size specified (BUDAGAVI; FULDSETH; BJONTEGAARD, 2014). Figure 2 shows an example of a quadtree division. The block in Figure 2(a) is a CTU representing the root node of the quadtree. The root node is split into four equal-sized nodes (1b to 4b) in Figure 2(b). In Figure 2(c), two of the four blocks (2b and 3b) are further split, and 1b and 4b blocks remain with the same size, turn into leaf nodes, and are converted into CUs. In the sequence, only the resulting blocks from 2b and 3b split (1c to 8c) are evaluated. Figure 2(d) shows that 1c, 4c, 6c and 8c blocks are not split and thus are converted into CUs, while blocks 2c, 3c, 5c, and 7c are further split. The resulting blocks from the last split (1d to 16d) are considered leaf nodes and converted into CUs since they have the minimum size allowed for a CU. Figure 2(e) shows the

quadtree representing the partitioning of CTU 1a.

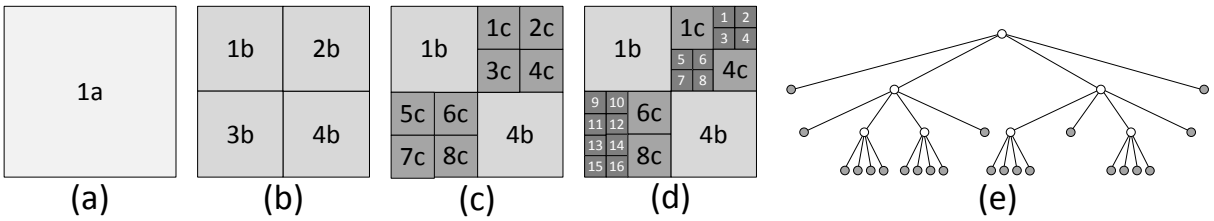

Figure 2 – Quadtree partitioning example

2.1.1.1 Coding Tree Unit

The CTUs are square-sized units with sizes of 64x64, 32x32, and 16x16. Nonetheless, different CTUs size ranges can be defined in the HM configuration files. The CTU size flexibility can be explored to adapt to the requirements of a specific application, such as memory restriction or coding/decoding time restriction. Usually, the use of bigger CTUs improves the coding efficiency by further reduction on the encoded video size (SULLIVAN et al., 2012).

2.1.1.2 Coding Units

The leaf nodes of CTU-rooted quadtrees are called CUs. The sizes allowed for CUs are 64x64, 32x32, 16x16, 8x8, and 4×4. Larger CUs are more appropriate to encode homogeneous regions. On the other hand, image regions with more details can be better encoded using a collection of smaller CUs. Thus, the bigger the image details level, smaller is the CU size recommended. Figure 3 shows the CU partitioning applied to a frame in the BasketballDrill video.

2.1.1.3 Prediction Units

Leaf nodes of a CU-rooted quadtree can be merged or combined into quadtrees having PUs as leaf nodes. Therefore, rectangular- or polygonal-shaped PUs are possible. Such formats are accomplished by recursively split CU-rooted quadtrees until the smaller unit size possible, and grouping adjacent leaf nodes, regardless their sizes to achieve such formats. However, to avoid unnecessary computational effort, some formats were predefined. The predefined sizes/formats for PUs and PBs are: 64x64, 64×48, 64x32, 64x16, 48x64, 32x64, 32x32, 32x24, 32x16, 32x8, 24x32, 16x64, 16x32, 16x16, 16x12, 16x8, 16x4, 12x16, 8x32, 8x16, 8x8, 8x4, 4x16, and 4x8. The prediction type in which the CU is used limits the type of allowed PU formats. Each CU can be sorted into three categories, according to the CU prediction type (SULLIVAN et al., 2012):

- *Intra Coded CU*: This category is applied to CUs used by the intra-picture prediction. A PB in an Intra Coded CU outputs a prediction direction (0 to 34) and

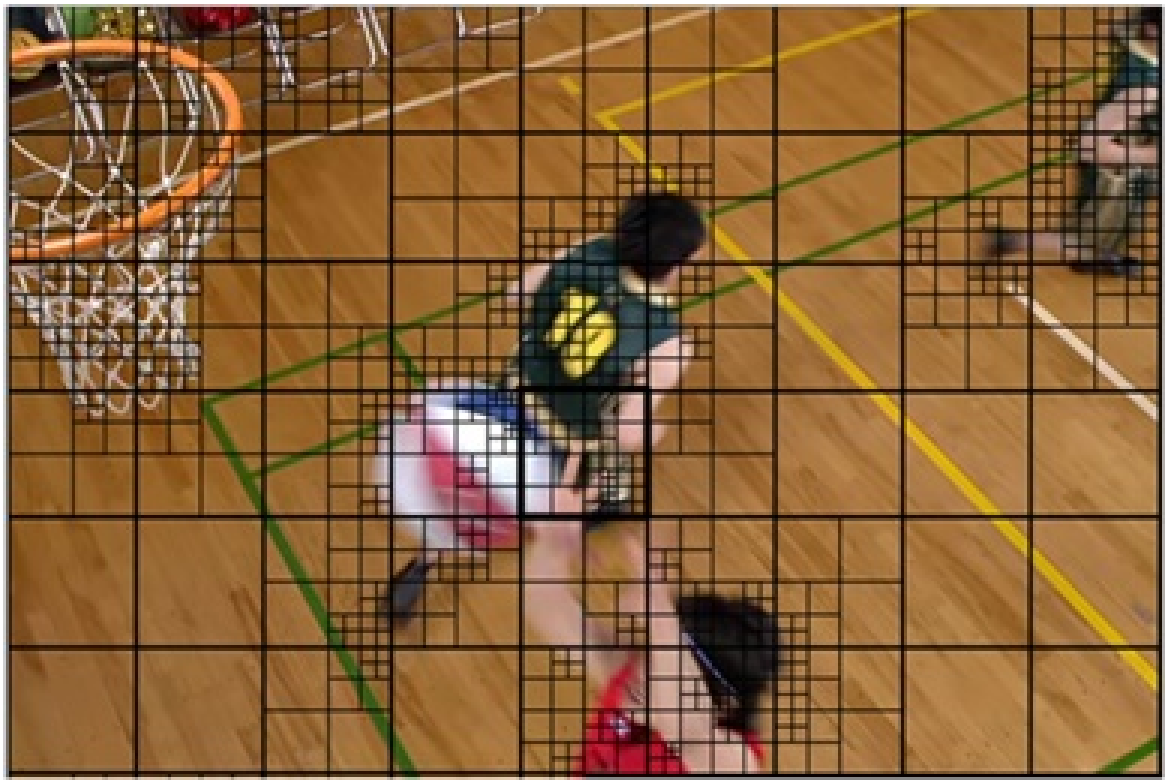

Figure 3 – CU partitioning applied to a frame in the BasketballDrill video (ZHOU; ZHOU; CHEN, 2013)

a residue block. Only square-shaped PUs (formats (a) and (b) in Figure 4) are used in Intra Coded CUs.

- *Inter Coded CU*: Applicable to CUs used by the inter-pictures prediction, its PB outputs a motion vector and a residue block. All predefined formats are allowed for Inter Coded CU PBs (formats (a) to (h) in Figure 4).
- *Skipped CU*: The Skipped CU category is a special case of the Inter Coded CU, where both the motion vector and the residue block are equal to zero. Skipped CUs cannot be split. Therefore, the respective PB must have the same size of the CU. The only format allowed for the PB is format (a) in Figure 4.

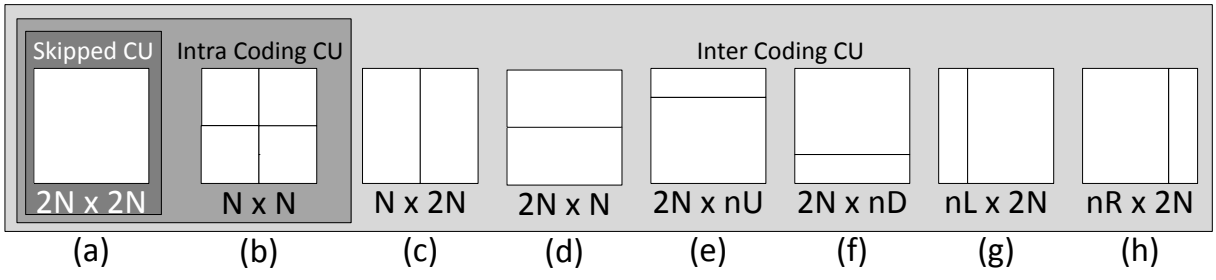

Figure 4 – PU formats allowed in HEVC

Frequently, chrominance PBs follow the partitioning scheme from the luminance PB in the same PU. However, when the CU size is 8×8 and the method of division is NxN, as in Figure 4(b), the 2Nx2N division method, in Figure 4(a) is used instead in the chrominance PB, avoiding a PU smaller than 4x8/8x4.

2.1.1.4 Transform Units

As it occurs with the PUs, one TU (or a set of TUs) is derived from each CU. The TUs are the leaf nodes of a CU-rooted quadtree, called Residual Quadtree (RQT). A TU is a unit containing residues in the spatial domain or quantized and transformed residues. In each TU, an integer discrete transform with the same dimension of the TU is applied to the residues in the spatial domain. The residues in the frequency domain, which are the transform outputs, are sent to the decoder after being quantized, in each TU (ITU-T, 2015). The residues in spatial domain can come from several PUs, which are regrouped into the CU used as root node for the TU quadtree (BUDAGAVI; FULDSETH; BJONTEGAARD, 2014). TUs were specified in four sizes: 32x32, 16x16, 8x8, and 4×4.

2.1.2 Residual Coding Loop

The HEVC RCL, the focus of this work, is composed of a direct transform, direct quantization, inverse quantization, and inverse transform. The RCL block diagram is shown in Figure 5.

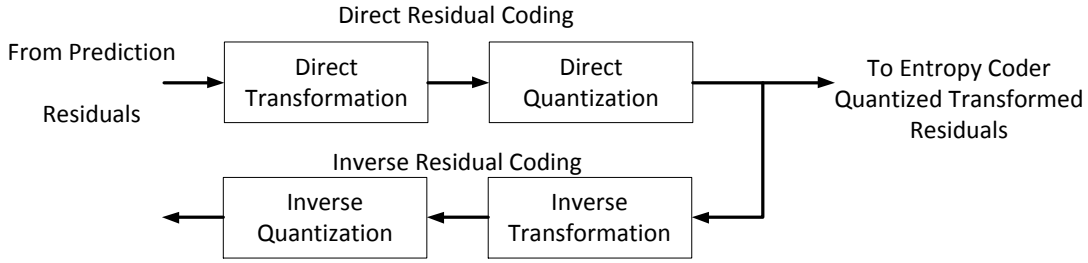

Figure 5 – RCL block diagram

Due to the characteristics of the residue information, the direct transform concentrates the relevant residue information in the low-frequencies related transformed residuals. On the other hand, most of the high-frequency related transformed residuals are orders of magnitude smaller than the low-frequency related transformed residuals. Direct quantization reduces the magnitude of the transformed residuals, reducing most of the quantized transformed residuals to zero, improving the entropy coding efficiency.

Figure 6 shows an example of the RCL using real data collected from HM. One can notice that 60 out of 64 quantized transformed residuals (QTR) values are zero, while the non-zero QTR values range from -3 to 4. This range of values eases the entropy coding because there is a large number of sequential bits with the same value.

The losses associated to the HEVC RCL are shown in the Error matrix, in Figure 6. The Error matrix is given by the difference between the original residues and the restored transformed residuals (RTR). Such errors are information losses introduced by the direct quantization.

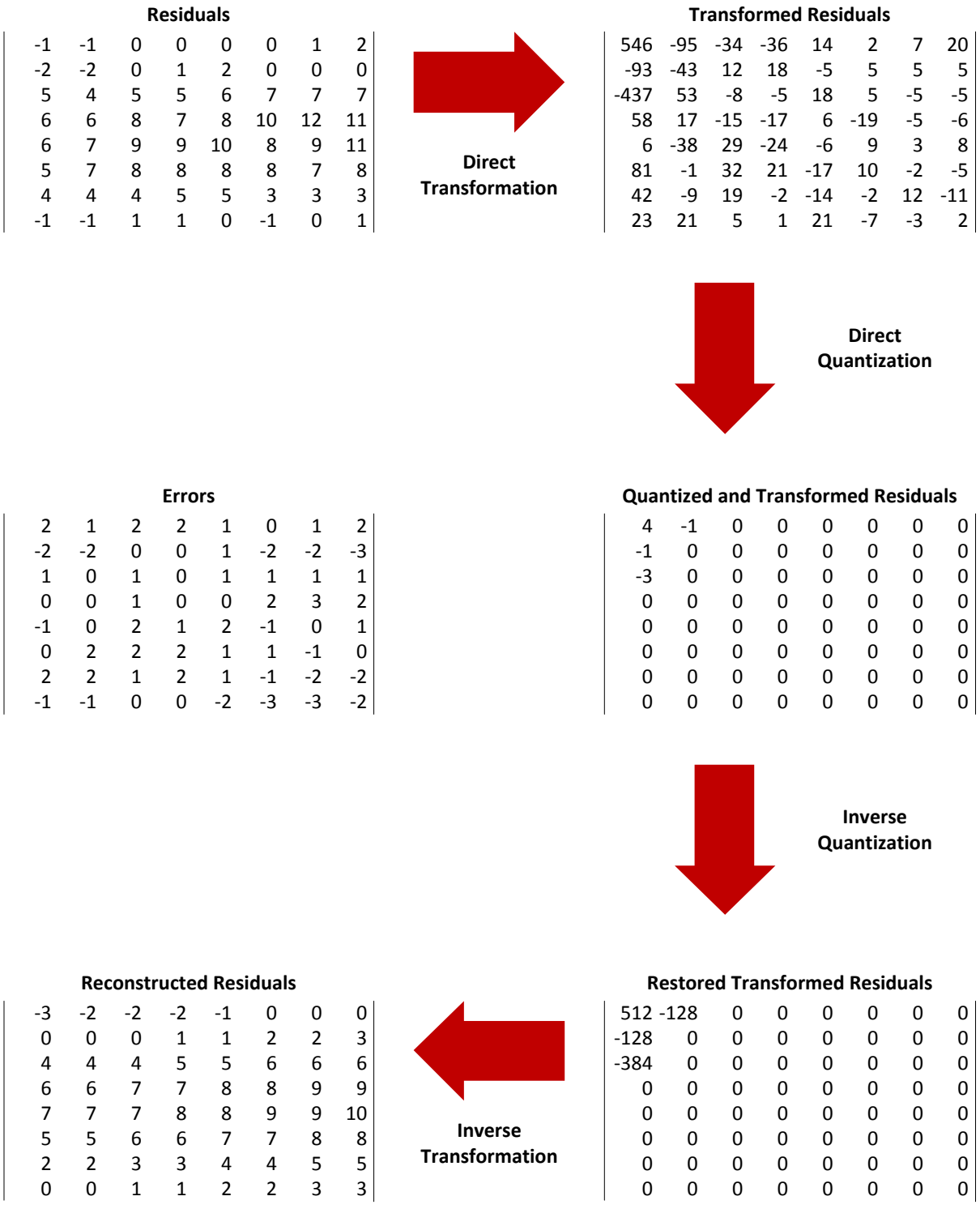

Figure 6 – Example of RCL operation

The quantization information losses are more apparent in the high-frequency Transformed Residuals (TR) data, since they are usually converted to zero in the direct quantization stage, making them unrecoverable. Nonetheless, those losses have a small impact on the perceptual quality of the reconstructed video image, since the Human Visual System (HVS) is less sensitive to high-frequency components of images (HAQUE; TABATABAI; MORIGAMI, 2011). Moreover, the losses inserted in the direct quantization stage can build up errors, as the images used in the encoder as reference are different from reconstructed images in the decoder. Therefore, the encoder implements and uses the inverse quantization and inverse transform to reconstruct the reference images as they are reconstructed in the decoder.

The HEVC RCL is used extensively during video encoding, since it must process different CM (combination of block sizes and prediction modes) to generate bitrate and distortion metrics, allowing the RDO to select the best trade-off between output encoded video size and reconstructed video image quality. However, there are a few special operation modes in HEVC (BUDAGAVI; FULDSETH; BJONTEGAARD, 2014) which affect RCL operation:

1. I_PCM Mode: The prediction, the transform, the quantization and the entropy encoding are not used in I_PCM mode. This mode is recommended when the residuals in one CU are not suiTable for hybrid coding processing, such as images with excessive noise. Such blocks processed by hybrid coding can generate an excessive quantity of bits. Therefore, the encoding processing is merely an adjustment on the output samples bit depth, when bit depth of input samples are different from bit depth of output samples.

2. Lossless mode: The transform and quantization are bypassed in this mode. Therefore, the decoder can reconstruct the video exactly as the original video. The lossless mode is defined at CU level, therefore, it is possible to mix lossy and lossless CUs in the same frame. The lossless mode can improve the coding of mixed-content videos, such as natural videos combined with text or synthesized images. In these cases, CUs with natural video data can use lossy encoding, while CUs containing synthesized elements can use lossless encoding.

3. Transform skipping mode: Using transform skipping mode, the transform is not used and the residuals on the spatial domain are applied directly to the direct quantization. This mode is used in cases where there are well-defined borders, such as synthesized images or texts. The use of transform and quantization in this type of images can deteriorate border definitions. Moreover, the transform skipping mode is restricted to 4×4 TBs.

The following subsections described the four components of the HEVC RCL:

2.1.2.1 Direct Transform

The HEVC specifies transform matrices with sizes 32x32, 16x16, 8x8, and 4×4. According to BUDAGAVI; FULDSETH; BJONTEGAARD (2014), more options in transform sizes allow bigger coding efficiency, although more transform sizes increase the computational effort.

The Discrete Cosine Transform (DCT) is the main transform method used in the HEVC and it is applied to all TB sizes. Alternatively, the Discrete Sine Transform (DST) can be used in intra-predicted luminance-residuals of 4×4 TBs. The two-dimensional DCT (2D-DCT) of size NxN produce transformed residue $F(u, v)$ from input residuals $f(u, v)$, as expressed in (2).

$$F(u,v) = \sqrt{\frac{2}{N}}\sqrt{\frac{2}{M}}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}\Lambda_i\Lambda_j cos\Big[\frac{\pi u}{2N}(2i+1)\Big]cos\Big[\frac{\pi v}{2M}(2j+1)\Big]f(u,v) \qquad (2)$$

where Λ_k = 1 and $\sqrt{2}$ for k=0 and k>0, respectively.

The transform operation through (2) uses a huge number of additions and multiplications, as seen in Table 1.

Table 1 – Arithmetical operations performed by 2D-DCT (JESKE, 2013)

Size	Additions	Multiplications
4 x 4	240	256
8 x 8	4,032	4,096
16 x 16	65,280	65,536
32 x 32	1,047,552	1,048,576

Since the transforms are always squared, i.e. $N = M$, and the way (2) is represented in (3), one can notice the DCT operation can be done in two steps (GHANBARI, 2003).

$$F(u,v) = \left\{\sqrt{\frac{2}{N}}\sum_{i=0}^{N-1}\Lambda_i cos\left[\frac{\pi u}{2N}(2i+1)\right]\right\} \times \left\{\sqrt{\frac{2}{M}}\sum_{j=0}^{M-1}\Lambda_j cos\left[\frac{\pi u}{2M}(2j+1)\right]\right\} \times f(u,v) \qquad (3)$$

Thus, the HEVC 2D-DCT can be processed through two executions of one-dimension direct transforms, using a buffer to store the partial results and transpose the coefficient matrix, according to (4) and (5), or in (6) using matrix notation. The transposition of the intermediate matrix f'(u,v) in (4) is denoted by the permutation of u and v indexes in (5).

$$f'(u,v) = \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} \Lambda_i cos\left[\frac{\pi u}{2N}(2i+1)\right] f(u,v) \tag{4}$$

$$F(u,v) = \sqrt{\frac{2}{M}} \sum_{j=0}^{M-1} \Lambda_j cos\left[\frac{\pi u}{2M}(2j+1)\right] f'(v,u) \tag{5}$$

$$F = C \times (f \times C)^T \tag{6}$$

The C_N matrices used in (6) are the transform matrices, where N denotes the size of the matrix. Basis vectors c_i of C_N, defined as $c_i = [c_{i0}, \ldots, c_{i(N-1)}]^T$, where $i = 0, \ldots, M-1$, are normal and orthogonal. Orthogonal vectors are vectors compliant to the property $c_i \times c_j^T = 0$ where $i \neq j$. Orthogonal vector allow output data be uncorrelated, which help increasing entropy efficiency. Since C_N basis vectors are normal vectors, compliant to the property $c_i c_i^T = 1$, a flat frequency response in the output transform matrix is assured.

Matrices C_N for blocks 32×32, 16×16, 8×8, and 4×4, are previously calculated and approximated to integer values. A straightforward manner to define the integer approximation for the matrix elements is to multiply each element for an integer number (usually between 2^5 e 2^{16}) and round them up to the nearest integer. Tables 28 to 31 in Appendix A represent the integer approximations matrices used by HEVC for the DCTs with block size 4×4, 8×8, 16×16, and 32×32, respectively.

Given the row indexes and column indexes ranging from 0 to $N-1$, the even-indexed rows in all matrices behave like even functions, i.e., the left half of the row (columns 0 to $(N-1)/2$) is mirrored in the right half of the row (columns $N/2$ to $N-1$). On the other hand, the odd-indexed rows behave like odd functions. Therefore, it is said that even-indexed rows present even symmetry, and odd-indexed rows present odd symmetry (BUDAGAVI; FULDSETH; BJONTEGAARD, 2014).

Based on these observations, HUNG; LANDMAN (1997) developed the Even/Odd Decomposition (EOD) algorithm. Using EOD algorithm, a 4-point 1D-DCT, shown in Figure 7, can be calculated using the following steps:

1. For each line, perform additions and subtraction in butterfly, according to Figure 7;
2. Multiply the previous results by the unique coefficients in Table 28 (36, 64, and 83);
3. Sum and subtract results from the previous step;
4. Adjust the bitlength in 1D-DCT output, rounding up to the nearest integer.

In step 4, the rounded bitlength adjust is performed by the addition of an offset value to the results from step 3, and a right shift operation. The offset value and the

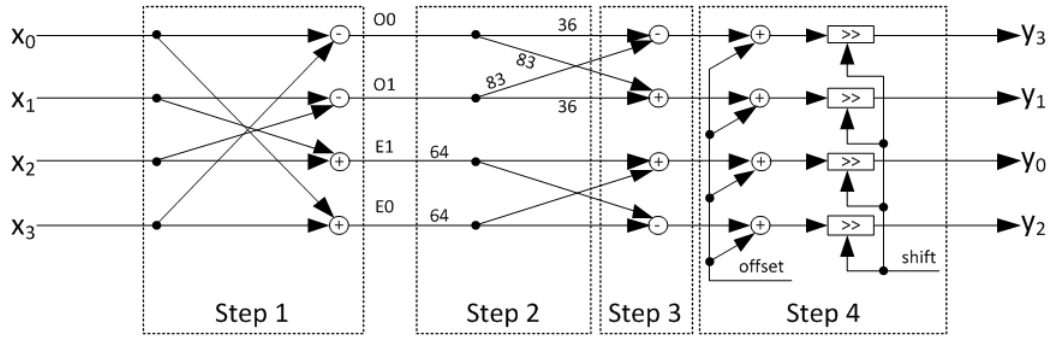

Figure 7 – 4-point 1D-DCT block diagram

Table 2 – Offset and shift parameters for different sized 1D-DCT (JESKE, 2013)

	1st 1D-DCT		2nd 1D-DCT	
DCT	offset	shift	offset	shift
4 x 4	1	1	128	8
8 x 8	2	2	256	9
16 x 16	4	3	512	10
32 x 32	8	4	1024	11

number of bits shifted in step 4 have different values for the first 1D-IDCT or the second 1D-IDCT and for different TB sizes. Table 2 shows offset and shift values for the four matrices sizes.

Moreover, the coefficients highlighted in Tables 29 to 31, in Appendix A, are the coefficients of the left-half of the even-indexed rows of the transform matrices, where the coefficients of any given transform matrix are exactly the coefficients of the immediate smaller transform matrix. For instance, the coefficients in the 4×4 transform matrix (Table 28) are the highlighted coefficients in the 8×8 transform matrix (Table 29). Therefore, in a hardware-implemented 1D-DCT, it is possible to reuse the hardware. The hardware reuse consists in using architectures of smaller 1D-DCT to process the samples of even-indexed rows of the immediate bigger 1D-DCT.

Direct Discrete Sine Transform

The HEVC standard can use a fixed point representation of Type-7 discrete sine transform (DST) to be applied to 4×4 luminance intra-prediction residual blocks. The forward DST transform matrix is given in Table 32, in Appendix A, although the inverse DST transform matrix is the transposition of the forward DST transform matrix. The usage of the DST provides around 1% bit-rate reduction while coding intra pictures(SAXENA; FERNANDES, 2013).

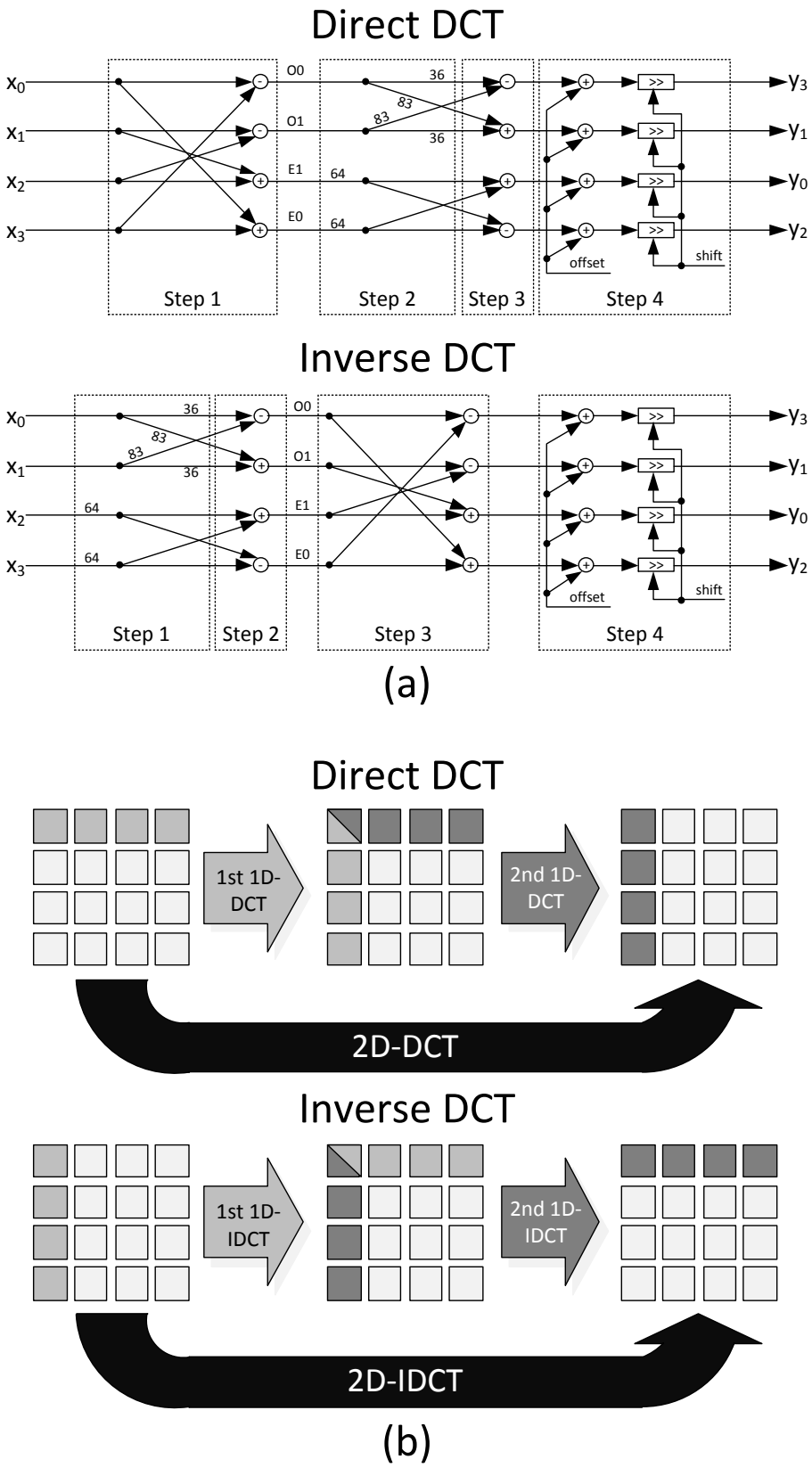

Figure 8 – Difference between DCT and IDCT. (a) EOD algorithm. (b) Processing order.

2.1.2.2 Inverse Transform

Inverse transform changes the representation of RTR from the frequency domain to spatial domain. The inverse transform is quite similar to the direct transform. Although, there are two main differences:

1. *Processing direction*: As shown in Figure 8, direct transform inputs (from the TB) are processed row-by-row and direct transform outputs are stored column-by-column. On the other hand, inverse transform inputs are processed by reading input TB data column-by-column and inverse transform outputs are stored row-by-row. Figure 8(b) shows the difference in processing direction between direct transform and inverse transform. Although inverse transform uses the same transform matrices C_N, the matrix notation of inverse transform is: $F = (C_N \times f) \times C_N^T$.

2. *Operation Order in EOD Algorithm*: A variation of EOD algorithm can be applied to the inverse transform, where the butterflies operations (stage 1 in Figure 7) occur after the weighted sum of RTR (stages 2 and 3 in Figure 7), as shown in Figure 8(a).

2.1.2.3 Direct Quantization

The quantization in the HEVC is performed as a function of the Quantization Parameter (QP) value, where it can assume values between 0 to 51. The QP value impacts on encoding time, output video size and reconstructed video quality. Bigger QP values are used to reduce output video size and encoding time, although they also reduce the reconstructed video quality. On the other hand, smaller QP values have less impact on the reconstructed video quality, despite the increase in the output file size and in the encoding time. Quantization consists in dividing transformed residuals by a Quantization Step (QStep) and rounding them to the nearest integer towards zero. The quantization is responsible for improving the conditioning of the transformed residuals and increasing the entropy coder efficiency. The quantization step value is determined by (7).

$$QStep = 2^{\frac{QP-4}{6}} \tag{7}$$

QStep values, as a function of QP, has a logarithmic characteristic, as shown in Figure 9. Point A in Figure 9 is one of many QStep values presenting fractional value. Processing fractional data requires high computational effort, therefore a simplification is required. On the other hand, point B and C, highlighted in Figure 9, show how QStep value doubles each time QP value increases of six. Therefore, QP can be divided by six, where the remainder of the division is used to determine the value of QStep, and

the quotient of the division indicates the number of bits to be shifted to right (equivalent to divide by 2^n).

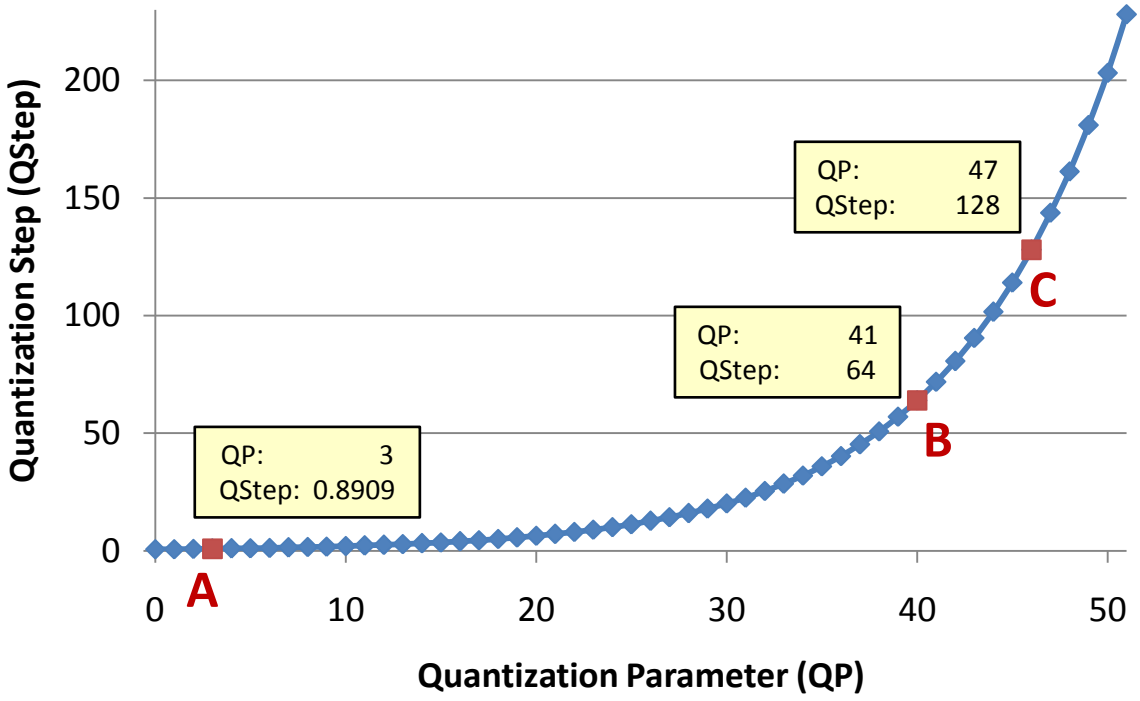

Figure 9 – Quantization step value according to QP value

Although the division-by-six approach can reduce the computational effort on the quantization, still a division by a fractional value must be processed. A simplified version of quantization is presented by BUDAGAVI et al. (2013), where the QStep values related to QP ranging from 0 to 5 are processed in advance and stored. Such QStep values are inverted and shifted 14 bits to the left. Therefore, an integer approximation of direct quantization is presented in (8), where $G = [26214, 23302, 20560, 18396, 16384, 14564]^T$ is the vector with inverse approximations of QStep related to QP ranging from 0 to 5 and $\lfloor . \rfloor$ is the rounding to the nearest integer towards zero.

The integer approximation for the direct quantization processing is given by (8), where:

- $qBits$, given by (9), is the number of bits that quantized transformed residue value must be shifted to the right to compensate the bitlength expansion of $G(QP\%6)$ and to adjust to the TB size (N);
- $offset$, given by (10), is a rounding correction parameter for the quantized transformed residue. Offset depends on the prediction type and the qBits parameter.

$$Z = sign(W) \times [|W| \times G(QP\%6) + offset] >> shift \tag{8}$$

$$qBits = 21 + \left\lfloor \frac{QP}{6} \right\rfloor - log_2 N \tag{9}$$

$$offset = \begin{cases} 171 \ll (qBits - 9) & \text{, if slice type is Intra-predicted} \\ 85 \ll (qBits - 9) & \text{, otherwise} \end{cases} \tag{10}$$

Since the HVS is less sensitive to high-frequency information, the HEVC can use the frequency-dependent quantization (FDQ) tool, which explores this HVS property. Therefore, FDQ usage reduces even further high-frequency transformed residuals and increases the coding efficiency. Moreover, (PRAGNELL; SANCHEZ, 2016) claim FDQ usage has a negligible effect on the perceptual quality of the reconstructed video. The QStep applied to quantized residues according to its frequency are given by (11), where u and v (ranging from 0 to 7) are the row and column position of the residue in the output transform matrix, respectively, and $f'(u, v)$ is the normalized spatial-radial frequency, in cycles per degree (PRAGNELL; SANCHEZ, 2016). Thus, the FDQ-complaint direct quantization equation is given by (12).

$$w(u, v) = 2,2 \times [0.192 + 0.114 \times f'(u, v)] \times e^{-[0.114 \times f'(u.v)]^{1,1}} \tag{11}$$

$$Z = sign(W) \times [|W| \times G(QP\%6) \times w(u, v) + offset] >> shift \tag{12}$$

The use of FDQ increases the number of quantization coefficients from six to 384, which could increase the memory access and energy consumption. The FDQ tool is optional in HEVC and disabled by default in its reference software (PRAGNELL; SANCHEZ, 2016).

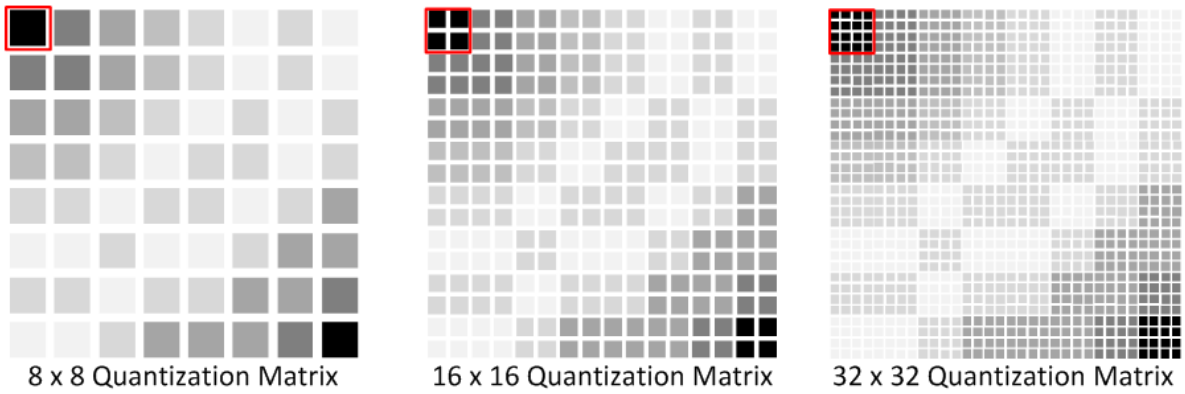

Figure 10 – Relationship between different QM sizes

Despite the HEVC support of TBs up to 32×32, only 8×8 quantization matrices (QM) are present in the HM (Tables 33 to 38 in appendix B), since the QM for 16×16 and 32×32 TBs are derived from the 8×8 QM. Each 8×8 QM coefficient is replicated twice in both directions to generate 16×16 QM. Similarly, each 8×8 QM coefficient is replicated four times in both directions to generate 32×32 QM. Figure 10 shows the coefficient relationship between 8×8 QMs, 16×16 QMs, and 32×32 QMs. There

are no 4×4 QMs since 4×4 TBs are not subjected to FDQ. Therefore, 4×4 TBs are processed as FDQ is disabled, regardless its configuration.

2.1.2.4 Inverse Quantization

The inverse quantization recovers the approximated TR values. It is achieved by multiplying the QTR by QStep, weighted according to the frequency if FDQ is enabled. The integer version of inverse quantization is given by (13) and (14) for the flat-inverse quantization and for the inverse quantization with FDQ support, respectively, where W' is the recovered transformed residue (RTR), Z is the quantized transformed residue, $h(QP\%6) = [40, 45, 51, 57, 64, 72]^T$ is an array complementary to $G(QP\%6)$, and $w(u, v)$ is the frequency-weighting function (11).

$$W' = \left\{ [(Z \times h(QP\%6)] << \left\lfloor \frac{QP}{6} \right\rfloor + [1 \ll (log_2 N - 2)] \right\} >> (log_2 N - 1) \tag{13}$$

$$W' = \left\{ \left[(Z \times \frac{h(QP\%6)}{w(u,v)} \right] << \left\lfloor \frac{QP}{6} \right\rfloor + [1 \ll (log_2 N - 2)] \right\} >> (log_2 N - 1) \tag{14}$$

In conclusion, the HEVC coder requires high computational effort due to the many encoding tools used to minimize the output encoded video file. Among such encoding tools, the RCL is highly required to provide information for the RDO process. Furthermore, HEVC encoding applications in video recording capable devices must encode videos in real-time to avoid storing huge amount of data between the image capture and the video encoding. Thus, HEVC RCL components must be implemented in hardware to achieve the required throughput for real-time video encoding.

2.2 Reference Software and Common Test Conditions

The HEVC Test Model Software (HM) is the reference software developed by the specialists of JCT-VC. The HM provides an HEVC standard implementation reference and serves as a basis to validate results from experiments seeking optimizations in the standard (FLYNN et al., 2011). Although the HM can be used as a reference for the generation of a valid HEVC bitstream, it was not developed for real-time encoding.

Along with the reference software, a Common Test Condition (CTC) was also defined (BOSSEN, 2013). The CTC is a collection of test video sequences and standardized configuration profiles which allow a fair, straightforward comparison between solutions using the CTCs. The CTC defined test video sequences (listed in Table 3) present different resolutions and content characteristics that must be considered for experimentations.

Along with the test video sequences, BOSSEN (2013) also defined four QP values

Table 3 – CTC Video Sequences (BOSSEN, 2013)

Class	Name	Resolution	Frame Rate	#Frames	Bitlength
A WQXGA	Traffic	2560x1600	30	150	8
	PeopleOnStreet	2560x1600	30	150	8
	NebutaFestival	2560x1600	60	300	10
	SteamLocomotive	2560x1600	60	300	10
B HD 1080p	Kimono	1920x1080	24	240	8
	ParkScene	1920x1080	24	240	8
	Cactus	1920x1080	24	240	8
	BQTerrace	1920x1080	60	600	8
	BasketballDrive	1920x1080	50	500	8
C FWVGA	RaceHorses	832x480	30	300	8
	BQMall	832x480	60	600	8
	PartyScene	832x480	50	500	8
D WQVGA	RaceHorses	416x240	50	500	8
	BQSquare	416x240	60	600	8
	BlowingBubbles	416x240	50	500	8
	BasketballPass	416x240	50	500	8
E HD 720p	FourPeople	1280x720	60	600	8
	Johnny	1280x720	60	600	8
	KristenAndSara	1280x720	60	600	8
F	BasketballDrillText	832x480	50	500	8
	ChinaSpeed	1024x768	30	300	8
	SlideEditing	1280x760	30	300	8
	SlideShow	1280x760	20	500	8

to be used in experimentations: 22, 27, 32, and 37. Moreover, there are four standardized configuration profiles for the HEVC encoder defined in the CTCs (BOSSEN, 2013):

- *All Intra* – The All-Intra configuration profile encodes all frames in the video using only intra-picture prediction. Moreover, frames are encoded in the same order they were recorded/generated. All-Intra configuration profile is the fastest configuration profile since an intra-picture prediction requires lesser computational effort than an inter-pictures prediction. On the other hand, All-Intra configuration profile presents lower compression rate than inter-frames encoding (BOSSEN, 2013).

- *Low Delay* – The Low-Delay configuration profile encodes all frames in the same order they were recorded/generated. In the Low-Delay configuration profile, the first frame is encoded using only intra-picture predictions. All the following frames allow inter-picture biprediction, using up to four reference frames (BOSSEN, 2013). Biprediction is achieved by combining two distinct intermediate inter-

picture predictions (SULLIVAN et al., 2012). The Low-Delay configuration profile has the advantage of high compression. However, it is not recommended for encoded videos to be broadcasted using Low-Delay configuration profile since information losses early in the video will propagate errors in the image up to its end.

- *Low Delay, P-frames only* – It is a variation of the Low-Delay configuration profile. In Low-Delay, P-frames only, configuration profile does not use bi predictions. Therefore, only one inter-frame prediction is made for each block in the frames following the first (BOSSEN, 2013). The Low-Delay, P-frames only, configuration profile present a small reduction in the compression rate, comparing to the Low-Delay configuration profile.

- *Random Access* – The Random-Access configuration profile divides the video into groups of 32 frames, by default. In each group, the first frame is encoded using only intra-picture predictions. The remaining frames in the group can be encoded using inter-pictures bi predictions. The inter-pictures predictions use only frames in the same group as references (BOSSEN, 2013). The Random-Access configuration profile encodes frames out of the order they are recorded/generated. The Random-Access configuration profile is the most appropriate to be used when coding videos for video broadcasting. It can reduce bitrate, by encoding most of the frames using inter-pictures prediction. On the other hand, the periodic intra-picture frame encoding purges errors in the video caused by missing data in the previous group of frames. Moreover, while using HEVC for digital-TV broadcasting using Random-Access configuration profile, the video decoder must wait until an intra-picture predicted frame is received, and can decode the following video. Using Low-Delay configuration profile, a TV show could be watched only if the TV set is turned on, or the channel is changed, at the exact beginning of the TV show.

3 HEVC RESIDUE CODING LOOP (RCL) ANALYSIS

This chapter presents experiments performed using HEVC reference software (HM) to evaluate the RCL throughput required to process all CM evaluated by the HM encoder, while another experiment evaluates the impact on the coding efficiency of RCL components hardware architectures throughput limitations. Moreover, the contribution of the frequency-dependent quantization (FDQ) tool to the HEVC coding efficiency is evaluated. All test video sequences in the CTCs (BOSSEN, 2013) were encoded while performing these experiments, where each video was encoded for QP values of 22, 27, 32, and 37, using All-Intra, Low-Delay, and Random-Access configuration profiles.

3.1 Throughput Requirements for the HEVC Residual Coding Loop

The purpose of the throughput requirement for the HEVC RCL experiment is to find how intensively RCL components are used during video encoding. The components in the RCL present symmetries between their direct modules and inverse modules, i.e. the throughput in the DCT component is the same throughput of the IDCT, as the throughput in the direct quantization is the same throughput of the inverse quantization. On the other hand, the throughput in the transform (DCT) can be different from the throughput in the quantization. The difference between the DCT throughput and the quantization throughput is due to the special RCL operational modes, as discussed in Section 2.1.2, or due to the use of the DST instead of the DCT in 4×4 TBs.

The RCL components are used several times during a CTB coding to evaluate different partitioning combinations and prediction modes. Therefore, the RCL throughput requirement analysis is based on the ratio between how many samples are processed by the RCL and the number of samples in the original video.

The RCL throughput requirement analysis is accomplished with a modification in HEVC Test Model, version 16.7. The modified HM records the number of residuals processed by direct DCT and direct quantization, by counting the number of TBs processed, for each size. The data retrieved from the modified HM determine the number

of residues processed by the RCL, individually tabbed for DCT and quantization. The RCL throughput is obtained individually for each video by the ratio between the data retrieved from the modified HM and the amount of data in the video.

The results, grouped by QP value, while averaging 19 videos in six resolution and three configuration profiles, and the overall average number of processed CM are presented in Table 4. The number of CM is calculated by the ratio between the number of processed samples and the number of samples of the original video. Due to the diversity of data averaged, the minimum and maximum number of CM for each QP value is also presented in Table 4.

Table 4 – Number of coding modes processed by RCL components, grouped by QP value

	DCT			Quantization		
QP	Minimum	Maximum	Average	Minimum	Maximum	Average
22	36.85	209.47	113.17	57.22	267.26	145.98
27	36.70	208.81	103.37	56.99	266.24	133.08
32	36.54	208.87	96.34	56.75	264.86	124.02
37	36.31	205.75	91.21	56.46	261.73	117.54
Overall	**36.31**	**209.47**	**101.02**	**56.46**	**267.27**	**130.15**

Table 4 shows an increase in the number of CM evaluated when QP value is reduced. Reducing QP value implies in higher image quality, and bitrate increase. Therefore, the encoder explores more intensively the CTB partitioning and prediction modes attempting to minimize the bitrate, which increases the number of CM evaluated by RCL components.

Table 5 groups the results according to video resolutions, with details of each video result, averaging the results of four QP values and three configuration profiles.

Table 6 details statistics from video resolution subtotals of Table 5 and from the overall dataset, including minimum, maximum, average, and standard deviation of the number of processed coding modes.

The chart in Figure 11 summarizes Table 6 subtotals and shows the average number of CM for the DCT and quantization, grouped by video resolutions.

The average number of CM, regarding the video resolutions, sits between 62.74 and 152.41 for the DCT, and between 83.34 and 192.26, for the quantization. Comparing the average values, it is noticeable the quantization being requested 30% more than DCT, in all video resolutions. Moreover, based on the number of tests performed (four QP values and three configuration profiles per video), the average by itself could not be enough to represent all gathered information. Therefore, the maximum and minimum information, for each resolution, are embedded to the chart in Figure 11, using error bars. Such error bars show a similarity among the minimum and the max-

Table 5 – Number of coding modes processed by RCL components, grouped by video and video resolution

Video	DCT			Quantization		
	Minimum	Maximum	Average	Minimum	Maximum	Average
BasketballPass	38.04	121.21	77.30	58.35	156.74	102.45
PeopleOnStreet	36.61	183.82	126.06	56.46	247.90	171.40
RaceHorsesC	38.65	162.89	100.57	59.23	215.35	132.69
WQVGA	**36.61**	**183.82**	**101.31**	**56.46**	**247.90**	**135.51**
BasketballDrill	40.46	156.68	90.69	60.79	198.99	117.81
BQMall	40.56	146.57	87.93	60.99	186.39	114.22
PartyScene	40.64	165.57	92.56	61.68	214.41	121.62
FWVGA	**40.46**	**165.57**	**90.39**	**60.79**	**214.41**	**117.88**
FourPeople	40.59	97.62	63.22	60.89	123.42	83.98
Johnny	40.14	95.45	61.83	60.27	120.89	82.23
KristenAndSara	40.48	97.52	63.17	60.65	123.17	83.81
HD 720p	**40.41**	**97.62**	**62.74**	**60.27**	**123.42**	**83.34**
BasketballDrive	50.97	155.85	93.11	60.03	193.60	114.73
BQTerrace	50.85	153.13	80.11	60.05	191.64	99.58
Cactus	51.35	140.68	85.07	60.50	176.07	105.37
Kimono	51.21	168.96	101.53	60.30	210.20	124.77
HD 1080p	**50.85**	**168.96**	**89.95**	**60.03**	**210.20**	**111.12**
NebutaFestival	38.28	199.36	117.85	60.81	267.27	162.70
Traffic	40.95	128.02	74.41	61.29	161.30	97.33
WQXGA	**38.28**	**199.36**	**91.78**	**60.81**	**267.27**	**123.47**
Beauty	49.38	209.47	153.58	60.75	266.03	192.97
ReadySetGo	49.39	207.69	151.86	60.59	263.58	191.54
ShakeNDry	49.34	206.51	152.25	60.36	216.85	191.54
YachtRide	49.42	206.32	151.96	60.13	261.58	191.54
UHD 4K	**49.34**	**209.47**	**152.41**	**60.13**	**266.03**	**192.26**

imum values and the average value. The chart in Figure 11 shows no correlation between video resolution and RCL throughput. Further analysis of the videos shows that the throughput is more related to the video characteristics than the resolution. The evaluated low-resolution videos (WQVGA and FWVGA) present high-mobility and heterogeneous-texture images, therefore the encoder must test more intensively different partitioning schemes and prediction modes, while the HD 720p videos have a static background; therefore, most of the CUs are skipped. Skipped CUs are not processed by the RCL. Thus, the RCL throughput for HD 720p videos is smaller.

The number of CM processed, grouped by configuration profiles, by averaging the four QP values and the 19 videos results, is shown in Table 7.

According to Table 7, the All-Intra configuration profile presents the lowest RCL number of CM. On the other hand, Low-Delay configuration profile presents the high-

Table 6 – Number of coding modes processed by RCL components, grouped by video resolution

Video Resolution	DCT				Quantization			
	Min.	Max.	Avg.	St.Dev.	Min.	Max.	Avg.	St.Dev.
WQVGA	36.61	183.82	101.31	13.75	56.46	247.90	135.51	25.87
FWVGA	40.46	165.57	90.39	87.68	60.79	214.41	117.88	88.75
HD 720p	40.41	97.62	62.74	12.61	60.27	123.42	83.34	16.15
HD 1080p	50.85	168.96	89.95	47.14	60.03	210.20	111.12	59.99
WQXGA	38.28	199.36	91.78	48.36	60.81	267.27	123.47	55.90
UHD 4K	49.34	209.47	152.41	102.00	60.13	266.03	192.26	130.44
Overall	**36.31**	**207.47**	**101.02**	**31.84**	**56.46**	**267.27**	**130.15**	**39.95**

est RCL number of CM. These results are mostly related to the number of intra-picture and inter-pictures predicted blocks in each configuration profile. Intra-picture prediction uses fewer PB sizes and formats and usually it has a limited number of directions evaluated. Therefore, intra-pictured predicted blocks require lower RCL throughput. On the other hand, inter-pictures prediction uses up to 24 PB sizes and formats to search for the best motion vector in each reference frame. Thus, inter-pictures prediction requires higher RCL throughput. The All-Intra configuration profile requires the least RCL throughput because it uses only intra-predicted blocks. The Low-Delay configuration profile has the majority of its blocks predicted using inter-pictures, only the first frame is mandatorily intra-picture predicted. Therefore, Low-Delay configuration profile has the highest RCL throughput requirement among the configuration profiles. The Random-Access configuration profile requires a periodicity of exclusively intra-picture predicted frames among the inter-pictures predicted frames. By default, such periodicity is one frame in every 32 frames. Therefore, the Random-Access configuration profile has a slightly lower RCL throughput requirement than Low-Delay configuration profile, and a higher RCL throughput requirement when compared to the All-Intra configuration profile.

Table 7 – Number of coding modes processed by RCL components, grouped by configuration profile

Configuration Profile	DCT			Quantization		
	Minimum	Maximum	Average	Minimum	Maximum	Average
All Intra	36.31	183.82	101.31	56.46	247.90	135.51
Low Delay	65.15	209.47	137.97	83.83	267.27	175.84
Random Access	61.87	204.76	121.88	80.35	259.77	155.20
Overall	**36.31**	**209.47**	**101.02**	**56.46**	**267.27**	**130.15**

The minimum working frequency for a hardware design of the RCL components

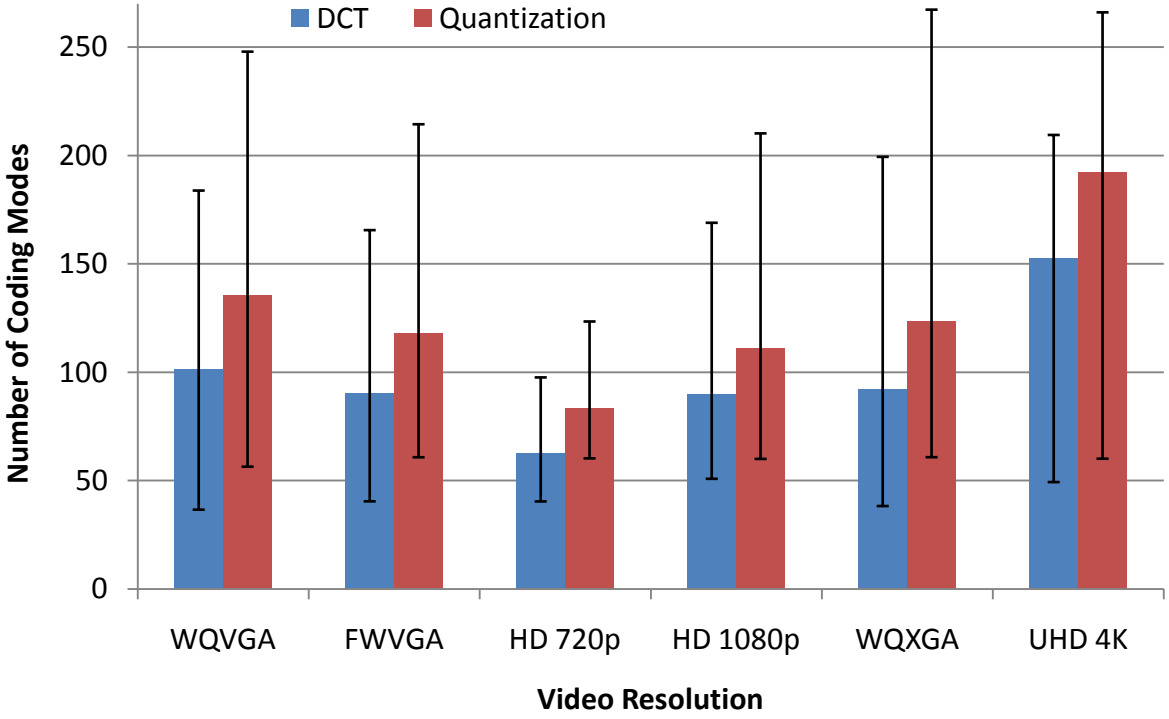

Figure 11 – Number of CM processed by RCL components, grouped by resolutions

is calculated based on its overall average throughput requirements: 101 for the DCT and 130 for the quantization. The minimum working frequency is calculated according to (15), where TR is the throughput requirement, w is the video width, h is the video height, pbw is the video pixel bitwidth, fr is the video frame rate and spc is the number of samples processed per clock cycle.

$$F_{min} = \frac{TR \times w \times h \times pbw \times fr}{8 \times spc} \tag{15}$$

Table 8 shows the minimum working frequency for hardware-designed DCT and quantization modules while processing 1, 4, 8, 16, or 32 samples per clock cycle. The number of samples per cycle is based on the TB sizes, considering that one line is processed per clock cycle, or single-sample processing for a serial processing. The single-sample processing could be used by quantization, despite the limited throughput. However, it should not be used by the DCT, since the DCT requires all columns data for processing.

Therefore, the minimum working frequency can be as high as $12.57GHz$ and $16.17GHz$, for the DCT module and the quantization module, respectively, when the modules process 32 samples per second. These frequencies are the required to fulfill the HEVC throughput demand to encode real-time UHD 8K videos at 120fps – the highest resolution defined nowadays.

Although desirable, many of those required frequencies are not practical for current

Table 8 – Minimum frequency for the DCT and quantization hardware modules, in GHz

Resolution	fps	Samples per cycle								
		DCT				Quantization				
		4	8	16	32	1	4	8	16	32
HD 720p	30	0.70	0.35	0.17	0.08	3,6	0.90	0.45	0.22	0.11
HD 1080p	30	1.6	0.79	0.39	0.20	8,1	2.0	1.0	0.51	0.25
WXQGA	30	3.1	1.6	0.78	0.39	16.0	4.0	2.0	1.0	0.50
UHD 4K	30	6.2	3.1	1.6	0.79	32.4	8.1	4.0	2.0	1.0
UHD 4K	60	12.6	6.3	3.1	1.6	64.7	16.2	8.1	4.0	2.0
UHD 8K	30	25.1	12.6	6.3	3.1	129.4	32.4	16.2	8.1	4.0
UHD 8K	60	50.3	25.1	12.6	6.3	258.8	64.7	32.4	16.2	8.1
UHD 8K	120	100.5	50.3	25.1	12.6	517.6	129.4	64.7	32.4	16.2

technologies, especially considering mobile devices. An early consensus among specialists predicted frequencies up to 10GHz for 45-nm manufactured general-purpose Central Processing Units (CPUs), while such CPUs were produced with a frequency between 1GHz and 4GHz (MARKOV, 2014). Multi-Processed Systems on Chip (MPSoCs) are very common in mobile devices. Such devices used to be composed of multiple processing units, such as CPU, Digital Signal Processor (DSP) modules, Graphical Processing Unit (GPU), and a set of hardware-implemented peripherals, such as Modem GSM/3G/4G, GPS modules, USB interfaces, image processing interfaces, multimedia processing interfaces (video encoding/decoding). Examples of current MPSocS are the QualComm Snapdragon 385 Mobile Platform (QUALCOMM, 2018), the Samsung Exynos 9 Series (SAMSUNG, 2018), and the Apple A10 Fusion (APPLE, 2017). These MPSoCs are currently manufactured using 10-nm technology and can achieve up to 2.9GHz (SAMSUNG, 2018). The maximum specified frequency is regarding the main CPU, where the other modules use to operate in lower frequencies, e.g. the main CPU in Samsung Exynos 9 Series (SAMSUNG, 2018) maximum frequency can be up to $2.9GHz$ while its GPU maximum frequency is limited to $710MHz$.

3.2 BD-Rate Evaluation of the RCL

DCT hardware implementations in related works (BONATTO et al., 2017), (MASERA; MARTINA; MASERA, 2017), (RENDA et al., 2017), (BASIRI; MAHAMMAD, 2017), (CHATTERJEE; SARAWADEKAR, 2017), (ANSARI; MANSOURI; AHAITOUF, 2016), and (JRIDI; MEHER, 2016) can achieve working frequencies up to $714.4MHz$, while quantization hardware-implementations (DIAS; ROMA; SOUSA, 2015) can operate up to $694.4MHz$. Such maximum frequencies are much lower than the minimum required to process UHD 8K videos in real time ($12.57GHz$ for DCT and $16.17GHz$ for quantization). As previously mentioned, such frequencies are required to ensure that

the hardware-implemented encoder is able to process all the CM processed by the reference software of the HEVC encoder. While processing all CM possibilities, the encoder can assure the highest level of coding efficiency. However, when there is a throughput limitation preventing the evaluation of all CM possibilities, a reduction in the coding efficiency is expected.

This experiment is performed by encoding three HD 1080p (BQTerrace, Cactus, and Kimono) and three UHD 4K videos sequences (Beauty, Jockey, and ShakeNDry), limiting TB partitioning tree depths. In the experiments, the Random Access configuration profile is used, and each video is encoded using the four recommended QP values (22, 27, 32, and 37) and four TB quadtree depths (32x32 to 4×4, 16×16 to 4×4, 8×8 to 4×4, and only 4×4). As mentioned in section 2.1.1.4, TBs are specified in four sizes: 32x32, 16x16, 8x8, and 4×4. Therefore, a full TB quadtree (32x32 to 4×4 TB quadtree depth) has a 32×32 TB as its root node and can have up to 4×4 TBs as leaf nodes. Nonetheless, in this experiment the size of the TB root nodes are reduced to 16x16, 8x8, and 4×4 while keeping the minimum leaf node size as 4×4. By imposing such reduction, there are fewer TB partitioning options to explore. Therefore, a reduction in the number of CM processed by the RCL components is expected.

The output encoded video size and the reconstructed video quality are evaluated and compared. The comparisons are evaluated using the Bjøntegäard distances (BJONTEGAARD, 2001) metrics. Such metrics measure the difference between two rate-distortion curves (R-D curves). These curves are plotted based on bitrate and PSNR values obtained from video encodings, using at least four different QP values. Therefore, CTC recommended QP values used were: 22, 27, 32, and 37.

The differences in bitrate between videos can be measured using BD-Rate (Bjøntegäard Distance on Bit Rate). The BD-Rate metric represents the increase in bit rate required for the evaluated video to produce a decoded video with the same image quality than the reference video. Therefore, a negative BD-Rate is advantageous, because implies that the method used can encode videos with the same quality as the reference, despite using fewer bits to represent it. Table 9 presents the resolutions average and the overall average BD-Rate metric and reduction in transform and in quantization number of processed CM. In Table 9, column BR(%) represents the BD-Rate variation of the current TB quadtree size range compared to the full TB quadtree size range (32x32 to 4×4), Tr(%) shows the reduction percentage on processed number of CM for the transform over the full TB size range, and Qt(%) shows the reduction percentage processed number of CM for the quantization over the full TB size range. The detailed information, including data from individual videos, can be found in the appendix C (Tables 45 and 46).

Analysis in Table 9 data, and revised with Tables 45 and 46 data, showed an adequate exponential approximations to BD-Rate impact regarding RCL throughput

Table 9 – Impact of reducing the TB size range on BD-Rate, number of processed coding modes of the DCT and the quantization

TB quadtree size range	QP	HD 1080p Videos			UHD 4K Videos			All Videos		
		Tr (%)	Qt (%)	BR (%)	Tr (%)	Qt (%)	BR (%)	Tr (%)	Qt (%)	BR (%)
4 x 4 to 4 x 4	22	54.41	36.60	24.37	48.34	31.04	3.95	51.38	33.82	14.16
	27	57.61	39.14	24.37	48.49	31.21	3.95	53.05	35.19	14.16
	32	58.80	40.48	24.37	48.71	31.44	3.95	53.76	35.96	14.16
	37	59.76	42.05	24.37	48.92	31.66	3.95	54.34	36.86	14.16
4 x 4 to 8 x 8	22	37.94	28.30	10.27	33.57	24.35	1.71	35.76	26.33	5.99
	27	41.11	30.81	10.27	33.75	24.53	1.71	37.43	27.67	5.99
	32	41.75	30.99	10.27	33.89	24.68	1.71	37.82	27.83	5.99
	37	42.11	31.09	10.27	34.05	24.84	1.71	38.08	27.96	5.99
4 x 4 to 16 x 16	22	18.83	15.07	2.77	16.61	13.09	1.12	17.72	14.08	1.95
	27	21.36	17.20	2.77	16.69	13.16	1.12	19.02	15.18	1.95
	32	21.79	17.47	2.77	16.78	13.26	1.12	19.29	15.36	1.95
	37	21.65	17.24	2.77	16.86	13.34	1.12	19.26	15.29	1.95

limitation, for both DCT component, given in (16), and quantization component, in (17), where $\#CM\_DCT$ is the number of CM processed by the DCT solution, and $\#CM\_Quant$ is the number of CM processed by the quantization solution. The coefficient of determination R^2 (DAVORE, 2011) for expressions (16) and (17) are 84.37% and 65.04%, respectively.

$$BdRate(\#CM\_DCT) = 59.37\% \times e^{-0.05 \times \#CM\_DCT} \tag{16}$$

$$BdRate(\#CM\_Quant) = 135.46\% \times e^{-0.07 \times \#CM\_Quant} \tag{17}$$

Figure 12 shows the BD-Rate impact from Tables 9, 45, and 46 and the impact tendencies according to the LCR number of processed CMs, using (16) and (17). The dots (red squares for the DCT and blue diamonds for the Quantization) in Figure 12 represent the data in Tables 9, 45, and 46, and the curves represent the trend lines of the impact of the number of processed CM on the DCT, in blue, and the Quantization, in red. According to the chart in Figure 12, the estimated impact on BD-Rate can be as high as 56.85% for the transform, and 125.87% for the quantization, for an LCR processing rate of one. Therefore, most of the related works have a big non-accounted impact on BD-Rate, due to its throughput limitation.

Table 10 summarizes the maximum frequency and number of samples per cycle processed by each of the related works, and its claimed BD-Rate impact, when applicable. Moreover, the processing rate in high-resolution videos is calculated based on the declared or calculated throughput and the BD-Rate impact is estimated , based

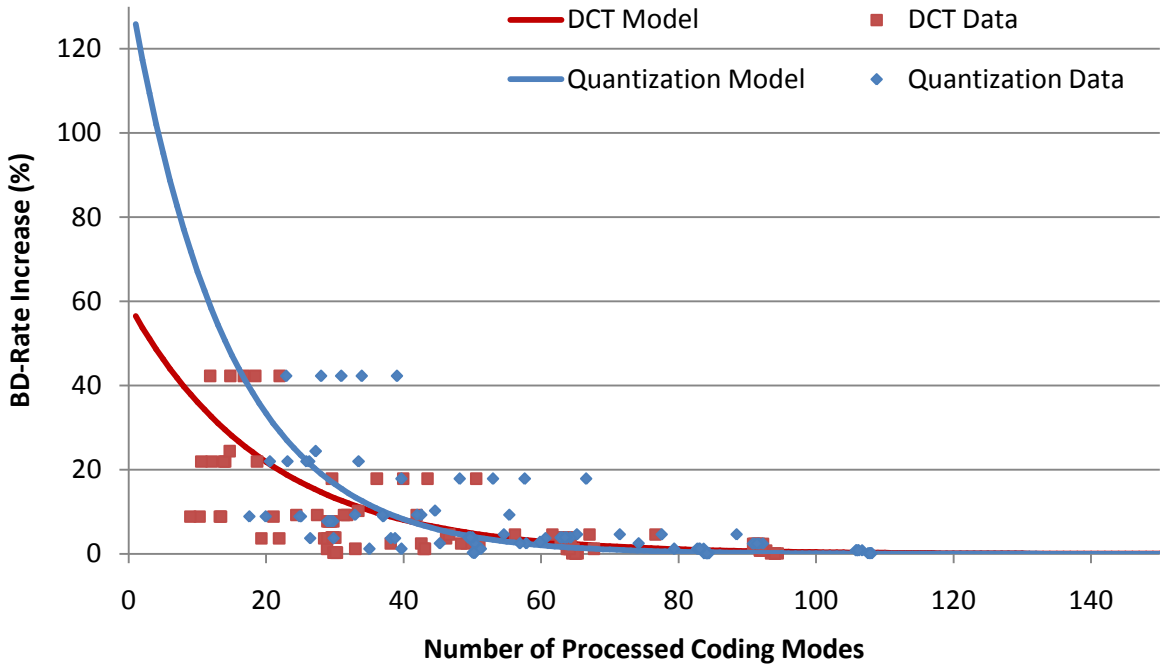

Figure 12 – BD-Rate impact tendencies according to the LCR components processing rates

on (16) and (17). In Table 10, throughput is given in samples per cycle (spc) and in giga samples per second (Gsps). Furthermore, the number of processed coding modes (#CM) and BD-Rate impact are calculated for two resolutions: HD 1080p (1920x1080 pixels) at 30 frames per second, and UHD 4K (3840x2160 pixels) at 60 frames per second. The BD-Rate impact ranges between 0.00% and 85.40% for the HD 1080p@30fps videos. For the UHD 4K@60fps videos, the BD-Rate impact ranges between 4.23% and 134.71%. Moreover, ANSARI; MANSOURI; AHAITOUF (2016) do not show enough performance to process UHD 4K@60fps videos in real-time. The BD-Rate estimate show significant reduction in the coding efficiency due to the RCL components throughput limitations. Therefore, hardware-implemented components for HEVC must provide high-throughput to reduce such BD-Rate impact.

Much of the HEVC RCL related works assume other HEVC components connected to them are capable to emulate the RDO process without using RCL. Since a coding-efficient RDO must consider the actual bitrate and distortion generated from each CM, RCL must process each CM. Therefore, a series of experiments show the number of CM processed by the HEVC RCL components: 101 for the DCT and 130 for the quantization. The number of processed CM reflect the ratio between the number of samples processed for RCL components and the number of samples in the video. Furthermore, the impact of limiting RCL components throughput over the size of the encoded video is evaluated. The model is derived from this evaluation shows an exponentially decreasing BD-Rate impact as the RCL components number of CM increase.

Table 10 – Related works estimated impact on BD-Rate due to processing rate limitations

Related Work	RCL Component	Maximum Frequency (MHz)	BD-Rate (%)	Throughput		HD@30fps		4K@60fps	
				spc	Gsps	Proc. Rate	BD-Rate (%)	Proc. Rate	BD-Rate (%)
RENDA et al. (2017)	DCT	324.00	4.74	32	10.36	111.11	0.00	13.89	19.08
JRIDI; MEHER (2016)	DCT	421.94	0.11	32	13.50	144.70	0.00	18.09	13.07
BONATTO et al. (2017)	DCT	185.50	-	6.84	1.27	13.60	19.58	1.70	57.14
MASERA; MARTINA; MASERA (2017)	DCT	250.00	-	32	8.00	85.73	0.03	10.72	25.38
ANSARI; MANSOURI; AHAITOUF (2016)	DCT	289.00	-	0.53	0.1541	1.65	57.38	0.20	-
BASIRI; MAHAMMAD (2017)	DCT	714.40	-	32	22.86	244.99	0.00	30.62	4.23
CHATTERJEE; SARAWADEKAR (2017)	DCT	145.10	-	8	1.16	12.44	21.73	1.55	57.88
DIAS; ROMA; SOUSA (2015)	Quantization	694.4	-	1	0.6944	11.16	85.40	1.40	134.71

Moreover, based on such model, the related works estimated impact on BD-Rate due to throughput limitations is shown, presenting estimated BD-Rate increases as high as 85.40% for HD 1080p@30fps videos and 134.71% for UHD 4K@60fps videos. Therefore, an effort must be made to achieve the highest throughput as possible for the RCL components, allowing more CM evaluation and reducing the coding efficiency losses.

3.3 Analysis of the Frequency-Dependent Quantization (FDQ)

As described in section 2.1.2.3, the HEVC supports the frequency-dependent quantization (FDQ). FDQ applies quantization steps (QSteps) according to the frequency associated with the transformed residue. PRAGNELL; SANCHEZ (2016) claim FDQ allows reduction in the encoded video size while having a negligible impact on restored image subjective quality.

Based on those premises, the intent of this experiment is to evaluate the impact of FDQ usage on the encoded video size and decoded video quality. Thus, the feasibility of including the FDQ in the RCL hardware architecture can be evaluated, mostly considering the trade-off between bitrate reduction and hardware area and power increase. For the current experiment, bitrate and PSNR information are logged from a series of encoded videos, following CTC (BOSSEN, 2013) specifications. Besides, a variation of CTC with FDQ enabled is executed where the bitrate and PSNR information are also logged.

The gathered information is evaluated using BD-Rate metric. Then the BD-Rate metrics are grouped according to the video and configuration profile where the configuration profile BD-Rates and the videos BD-Rates have their statistics extracted, respectively.The BD-Rate data are grouped by video resolution, in Table 11, and by configuration profile, in Table 12. The Tables present the grouping category, Resolution for Table 11 and Profile for Table 12, respectively. The Minimum column and Maximum column show, respectively, the least and the biggest BD-Rate value among the BD-Rate values fallen into each category, the Average column shows the average of the BD-Rate values of each category, and the Standard Deviation columns shows the standard deviation in the BD-Rate dataset in each category.

In Table 11, UHD 4K videos (3840 x 2160 pixels) stand out because they present an average bitrate reduction of 2.15%, achieving up to 7.94% bitrate reduction. On the other hand, the average bitrate reduction for videos with resolution up to WQXGA (2560 x 1600) is less than 1%, with maximum 2.85% bitrate reduction. The average data in Table 10 supports the recommendation of using FDQ for UHD videos. However, it is noticeable a broad variability in BD-Rate values for UHD resolution, since the standard deviation (2.10%) is close to the average results (-2.15%).

Table 11 – FDQ usage impact on BD-Rate (in %), grouped by video resolution

Resolution	BD-Rate (%)			Standard Deviation
	Minimum	Maximum	Average	
WQVGA	-2.85	0.24	-0.70	0.90
FWVGA	-1.38	-0.25	-0.87	0.33
HD 720p	-0.92	0.02	-0.53	0.32
HD 1080p	-2.34	0.64	-0.33	0.86
WQXGA	-0.77	-0.32	-0.54	0.18
UHD 4K	-7.94	-0.32	-2.15	2.10
Overall	**-7.94**	**0.64**	**-0.92**	**1.28**

Table 12 – FDQ usage impact on BD-Rate (in %), grouped by configuration profile

Resolution	BD-Rate (%)			Standard Deviation
	Minimum	Maximum	Average	
All Intra	-7.94	-0.06	-1.58	1.89
Low Delay	-1.62	0.64	-0.52	0.54
Random Access	-2.38	0.52	-0.65	0.67
Overall	**-7.94**	**0.64**	**-0.92**	**1.28**

Table 12 presents the BD-Rate metrics grouped by configuration profiles. The All Intra configuration profile shows the largest BD-Rate reduction among the configuration profiles. However, the All Intra configuration profile data is very variable, as it occurs in UHD resolution in Table 10. For the All Intra configuration profile data, the standard deviation (1.89%) is larger than the average bitrate reduction (1.58%). Moreover, the All Intra configuration profile produces the largest encoded video file among all the configuration profiles.

On the other hand, FDQ is not as hardware-friendly as the flat-quantization (non-frequency-dependent quantization). Flat-quantization in (8) requires multiplying TR data by one among six previously calculated inverse QSteps, while FDQ in (12) requires one more multiplication by a frequency-dependent weight. Therefore, flat-quantization requires one multiplication and a ROM for storing the six previously calculated inverse QSteps. On the other hand, FDQ requires two multiplications and a ROM for storing the six previously calculated inverse QSteps and the 8×8 frequency-dependent weights matrix. Alternatively, FDQ can require one multiplication and a ROM for storing a 6x8x8 3D-matrix combining the six previously calculated inverse QSteps and the 8×8 frequency-dependent weights matrix.

3.4 Final considerations

The HEVC coder minimizes the output encoded video bitrate through an extensive search for the most similar block among the previously encoded data. Such search

pre selects several CM to be processed by the RDO process, which uses the RCL to provide the output bitrate and image distortion of the blocks. For instance, the RCL processing rate experiment, in section 3.1, processed 39,204,864,000 (39.20×10^9) original video samples, where the DCT module processed 3,960,566,747,818 (3.96×10^{12}) samples and the quantization processed 5,102,651,011,516 (5.10×10^{12}) samples. Thus, the DCT and the quantization processed on average 101.02 and 130.15 more samples than the number of samples on the original videos, respectively. Therefore, the RCL components should present enough throughput to ensure real-time video encoding.

Nonetheless, hardware-implemented RCL related works usually consider processing the one prediction modes, which can affect negatively the bitrate outcome. Section 3.2 shows that most of RCL related works claiming UHD 4K real-time can process between 1.40 and 30.62 CM while encoding UHD 4K@60fps, which have an estimated BD-Rate increase between 4.23% and 134.71%.

Therefore, energy-aware, coding-efficient RCL hardware designs must optimize the trade-off between such antagonistic characteristics. The HEVC RCL solutions developed in this master thesis approach this challenge using parallelism to increase the throughput while reducing the operational frequency and using fine-tuned approximation techniques. The hardware simplification provided by the approximation technique improves the throughput in order to compensate the BD-Rate increase introduced by the approximation technique through increasing the number of processed CM.

4 HARDWARE SOLUTIONS FOR THE RESIDUAL CODING LOOP (RCL) COMPONENTS

Mobile devices must present the expected performance while saving their battery as long as it is possible. Video encoding is an intensive processing operation which can benefit from hardware accelerators. Therefore energy-saving techniques must be applied in the hardware design for video coding in mobile devices. Hardware accelerators implementing intensive processing algorithms are very common in Multiple-Processed Systems-on-Chip (MPSoCs) used in mobile devices. Such hardware accelerators can process information faster and more energy-efficiently than a software solution running in one of the MPSoCs processors. Moreover, the hardware accelerators can be power gated, i.e. can be turned off when they are not being used.

As demonstrated in chapter 3, the HEVC encoder version implemented in the HM maximizes the reference space exploration to improve the coding efficiency, regardless the encoding time. Section 3.1 shows HM transforms with an average of 101 coding modes for each CTB and quantizes an average of 130 coding modes for each CTB. The encoded CTB size and reconstructed CTB image quality for each coding mode are compared and the best trade-off between encoded size and reconstructed image quality is used for the video encoding.

Current mobile devices often present limited storage availability. Therefore, such devices must encode videos in real-time to reduce the amount of data to be stored. Table 8, in section 3.1, show the minimum frequency requirement for real-time processing of beyond-high-definition videos. The frequencies required can be up to $12.6GHz$ and $16.2GHz$, for the DCT and quantization hardware architectures, respectively. These frequencies are the required to process 32 samples per cycle of UHD 8K (7680 x 4320) at 120 frames per second. Targeting UHD 4K videos at 60 frames per second, the frequencies of $1.6GHz$ and $2.0GHz$, for the DCT and quantization hardware architectures, respectively, are more feasible although still high.

In this chapter, many solutions for the RCL components are proposed, as presented in Figure 13.

An FFT-inspired 1D-DCT is presented in section 4.1. Section 4.2 shows an qual-

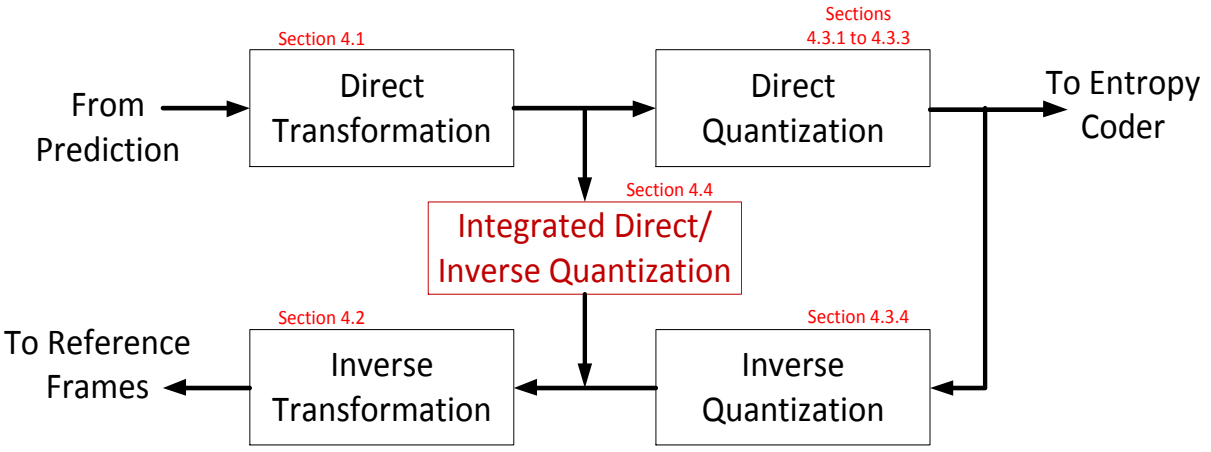

Figure 13 – RCL diagram block

ity/energy scalable solution for the inverse 1D-IDCT. Different variations of the direct quantization and inverse implementation are explored in section 4.3. Furthermore, an integrated direct/inverse solution is presented in section 4.4.

Several works related to RCL components process up to 32 samples per cycle (RENDA et al., 2017) (JRIDI; MEHER, 2016) (MASERA; MARTINA; MASERA, 2017) (BASIRI; MAHAMMAD, 2017) (BRAATZ et al., 2017). Such parallelism is based on the 32×32 TB maximum size.

The TB sizes available in HEVC are 4×4, 8x8, 16x16, and 32x32. Moreover, the biggest TB size (32x32) is often used as the root node of a quadtree. Therefore, the hardware architectures for the RCL components use parallelism to process 32 samples per cycle. The set of samples processed per cycle can be one line of a 32×32 TB or one line of a combination of smaller TBs. Thus, the RCL components hardware architectures are able to handle any possible partitioning decision in the TB quadtree, always processing 32 samples per cycle.

4.1 FFT-Inspired HEVC 1D-DCT Hardware Design

The HEVC 1D-DCT allows hardware reduction by using the Even-Odd Decomposition (EOD) algorithm (HUNG; LANDMAN, 1997), and by the reuse of smaller size 1D-DCT blocks to process the even-indexed columns of immediate bigger 1D-DCT data. On the other hand, such hardware reduction approach limits the 1D-DCT throughput since it can process data from only one TB size per clock cycle. The 32 samples per cycle throughput, defined in the beginning of this chapter, can be achieved by implementing one 32-point 1D-DCT block, one 16-point 1D-DCT block, two 8-point 1D-DCT blocks, and four 4-point 1D-DCT blocks. This way, it is possible to process lines from any TB partitioning combination by selecting the proper 1D-DCT blocks. For instance, when one line composed of lines from two 16×16 TBs is processed using the 16-point

1D-DCT block and the even part of the 32-point 1D-DCT block. On the other hand, the four 4-point 1D-DCT blocks, two 8-point 1D-DCT blocks, and the hardware responsible for processing the odd part of the 32-point 1D-DCT block are not used while processing two 16×16 TBs lines.

Despite the use of clock-gated registers can reduce the dynamic power dissipation, the static power dissipation of such blocks are still considerable for the overall power dissipation. In addition, the use of clock-gated registers increases the static power dissipation and can diminish the dynamic power dissipation reduction, since the registers switching dissipates more power than the combinational logic while transitioning. Thus, a 1D-DCT solution inspired on the Cooley-Tukey Fast-Fourier Transform (FFT) algorithm (COOLEY; TUKEY, 1965) is designed to further improve hardware reuse. The radix-2 decimation-in-time (DIT) FFT is the simplest, most computationally efficient and most common form of the Cooley-Tukey algorithm (JONES, 2016).

4.1.1 Radix-2 DIT FFT algorithm

The FFT algorithm was developed to reduce the computational complexity of the Discrete Fourier Transform (DFT) from $O(N^2)$ to $O(Nlog_rN)$, where r is the FFT DIT radix.

Radix-2 DIT FFT algorithm divides an N-sized DFT into two interleaved DFT of size N/2, at each recursive stage. Such division is accomplished by recursive even/odd separation of the current input data. The recursive even/odd separation is known as bit-reversal ordering (BRO). Moreover, the FFT algorithm splits one DFT with size N into N DFTs with size one. Then, the FFT algorithm groups the N DFTs results in the reverse order of the division until the whole DFT is calculated (JONES, 2016).

4.1.2 Adjusting the radix-2 DIT FFT for application in HEVC DCT

The main difference between the FFT and the HEVC DCT is the numeric set used, where the FFT computes the transform using a complex numbers set and the HEVC DCT using only integer numbers. Moreover, the DCT is a special case of the FFT where the imaginary values are discarded. Furthermore, the arithmetic operations using integer numbers are less computationally demanding than the ones using complex numbers. Therefore, an optimal hardware architecture can be designed where not only the number of operations is reduced, as in the FFT, but operates only with integer numbers, as in the HEVC DCT.

The DCT can be calculated by taking the real part of the rotation of the complex DFT results (PLONKA; TASCHE, 2005). In this case, the complex number multiplications and additions are mainly responsible for increasing the number of operations. On the other hand, SELESNICK; SCHULLER (2000) shows that the DFT can be simplified using only real-valued data by extending the input data sequence to condition the data

to the required symmetry, which allows DFT processing using only real values. The setback in such approach is that it requires processing a 2N-sized DFT for an N-sized input data. On the other hand, the real DFT algorithm (SMITH, 1997) can calculate real DFT with a reduced number of computations due to a data pre-processing and data post-processing. The real DFT algorithm forces the imaginary part of the input data to be zero and the real part of the input data to have even-symmetry. The algorithm splits the N-point input data, where the even-indexed input data are used as the real part of the DFT input and the odd-indexed input data are used as the imaginary part of the DFT input. Therefore, a N/2-sized FFT is required to process both halves of the N-point input data. After calculating the complex FFT, the frequency spectra are separated using an even/odd decomposition. Finally, the last recursion of the FFT is executed. Figure 14 shows a block diagram of a 32-point real DFT, using radix-2 DIT FFT algorithm, where the EOS (even/odd separation) blocks split the input data into even- and odd-indexed input data, the BC (butterfly cluster) blocks groups the even- and odd-indexed input data by calculating weighted butterflies, and the EOD (even/odd decomposition) take the output of two previous BCs, concatenates them and decomposes them into signals with even symmetry and with odd symmetry signals.

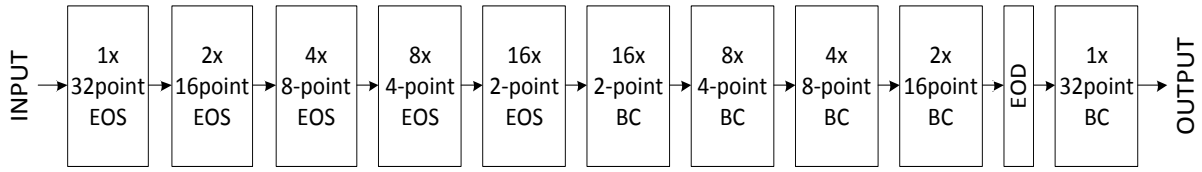

Figure 14 – A 32-point real DFT block diagram, using radix-2 DIT FFT

The HEVC DCT can be obtained by rotating the real DFT results, as in (COELHO; CINTRA; DIMITROV, 2017). The rotation is shown in (18), where Y_n is the n-th DCT output, X_n is the n-th real DFT output, N is the input data size, and n is the output index. Such rotation can be simplified since all elements of X_n are real numbers (19), i.e. their imaginary part are equal to zero.

$$Y_n = \mathcal{R}\left\{e^{-\frac{j\pi n}{2N}} \times X_n\right\} = \left\{cos\frac{\pi n}{2N} \times \mathcal{R}\left\{X_n\right\} + sin\frac{\pi n}{2N} \times \mathcal{I}\left\{X_n\right\}\right\} \tag{18}$$

$$Y_n = \mathcal{R}\left\{e^{-\frac{j\pi n}{2N}} \times X_n\right\} = cos\frac{\pi n}{2N} \times \mathcal{R}\{X_n\} \tag{19}$$

4.1.3 FFT-inspired HEVC 1D-DCT hardware architecture

Thus, the proposed HEVC DCT diagram block is shown in Figure 15. The differences between Figure 14 and Figure 15 are the Output Rotation block, which implements (19), and the insertion of optional EOD blocks before each 4-, 8-, 16-, and 32-point BC block. Since one of the design constraints is a 32-samples per clock cy-

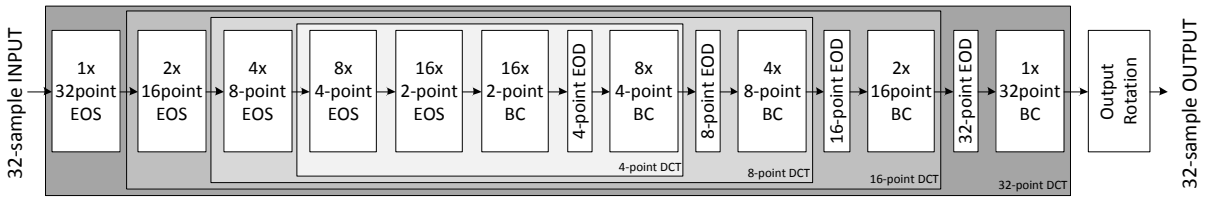

Figure 15 – Proposed DCT block diagram

cle processing ability regardless the TB partitioning structure, the EOS blocks, EOD blocks and BC blocks of each TB size can be bypassed and disabled according to the TB partitioning structure.

When processing a line composed of eight 4-point data, the 32-, 16-, and 8-point EOS blocks are bypassed, while the 4- and 2-point EOS blocks are used to apply the bit-reverse ordering of the 4-point data. The bit-reverse ordered 4-point data are processed by the 2-point BC blocks, resulting in the intermediate real DFT results, which have their real and imaginary parts concatenated and decomposed into signals with even and odd symmetry by the 4-point EOD blocks. Next, the even-symmetric signal from the 4-point EOD blocks is processed by the 4-point BC in the final stage of the real DFT. When processing 4-point blocks, subsequent 8-, 16-, and 32-point EOD and BC blocks are bypassed. The 4-point 1D-DCT outputs are obtained by applying the output rotation in the outputs from the 4-point BC blocks.

In contrast, when the proposed DCT design is processing one line from a 32×32 TB, all of the EOS blocks are used to perform the bit-reversal ordering. Moreover, the 2-, 4-, 8-, and 16-point BC blocks are responsible for the intermediate FFT calculation, while the 4-, 8-, and 16-point EOD blocks are bypassed. Thus, the 32-point EOD and BC blocks process the data in the final stage of the real 32-point DFT. At last, the real 32-point DFT outputs are rotated in the Output Rotation block, resulting in the 32-point 1D-DCT output.

The FFT-inspired HEVC 1D-DCT hardware design disables and bypasses blocks of the Figure 15 according to CONFIG input, although different actions are taken according to the type of disabled/bypassed block. Since the EOS blocks do not require arithmetical or logical operations, they can be bypassed only using a multiplexer in the output selecting the input if the EOS block is disabled or the EOS output if EOS block is enable, as shown in Figure 16(a). On the other hand, the computational effort requirement on BC blocks and EOD blocks makes more difficult to perform an energy-efficient bypass. By using the same approach as in the EOS blocks, the data toggling in the inputs of bypassed BC blocks or EOD blocks can cause the internal nets in such blocks to switch. Thus, dynamic energy is consumed while not producing usable results. Therefore, the clock-gated bypass, shown in Figure 16(b), is used for BC blocks and EOD blocks. The clock-gated bypass uses two clock-gated registers on the inputs

of the block. When the block is bypassed, the register A in Figure 16(b) stores the input data to forward it to the output multiplexer. On the other hand, the register B in Figure 16(b) stores the input data and forwards it to the operational block - Data path in Figure 16(b), when the bypass is disabled. Therefore, the blocks outputs are the output of the operational block.

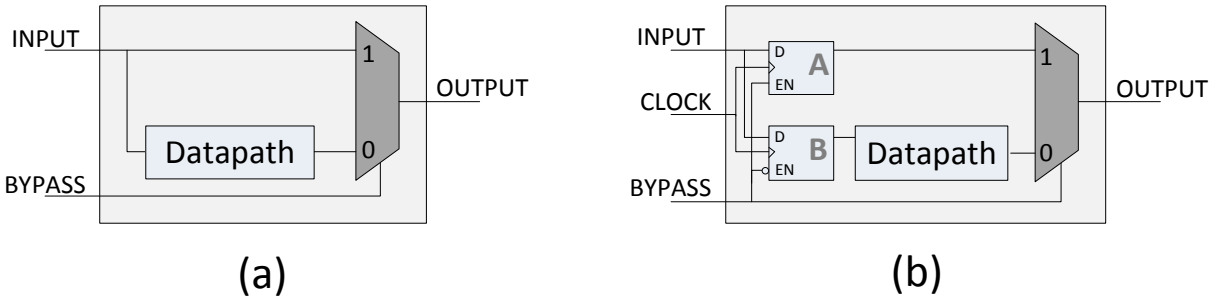

Figure 16 – Bypass approaches. (a) Simple-multiplexed bypass (b) Clock-gated bypass

The blocks in Figure 15 are detailed in the next subsections:

4.1.3.1 Even/Odd Separation (EOS) blocks

The EOS blocks split its input data based on the sample index, where the even-indexed input samples are grouped in one of its output and the odd-indexed input samples are grouped in the another output. Figure 17 shows the example of a recursive one 16-point data separation into 16 1-point data. The recursion is performed in four steps.

4.1.3.2 Butterfly Cluster (BC) blocks

The BC blocks is composed of a typical FFT weighted butterfly, shown in Figure 19(a). Each individual BC block in Figure 15 uses a different complex number constant for S in Figure 19(a). However, the S constants are converted into two fixed point numbers, which are converted into sets of additions and shifts. After the multiplierless multiplications, the butterfly operation is performed and a final 8-bit right shift is performed to compensate the fix-point operation. Moreover, the BC block size interposes the inputs differently from each other, as shown in Figure 18.

4.1.3.3 Even-Odd Decomposition (EOD) blocks

The EOD blocks decompose the input signal into one signal with even symmetry and one signal with odd symmetry, by using (20) and (21), respectively. Therefore, as shown in Figure 19(b), the EOD blocks are composed of a butterfly whose outputs are right shifted by one bit.

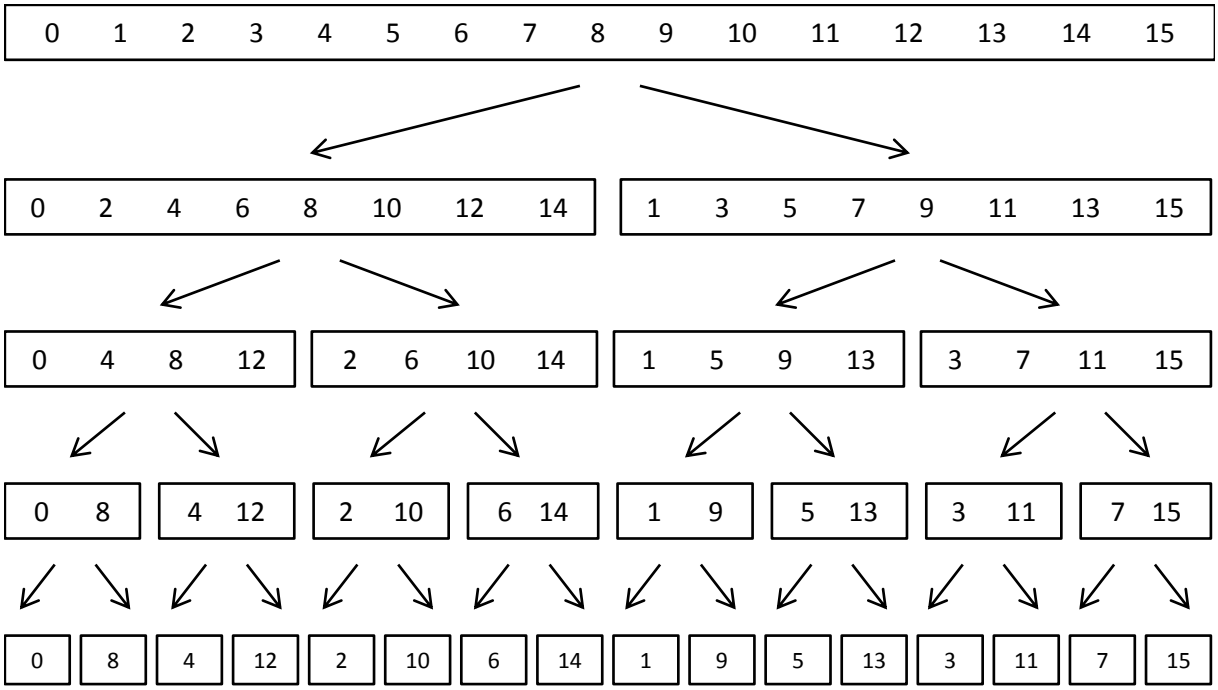

Figure 17 – Recursive 16-point even/odd separation (SMITH, 1997)

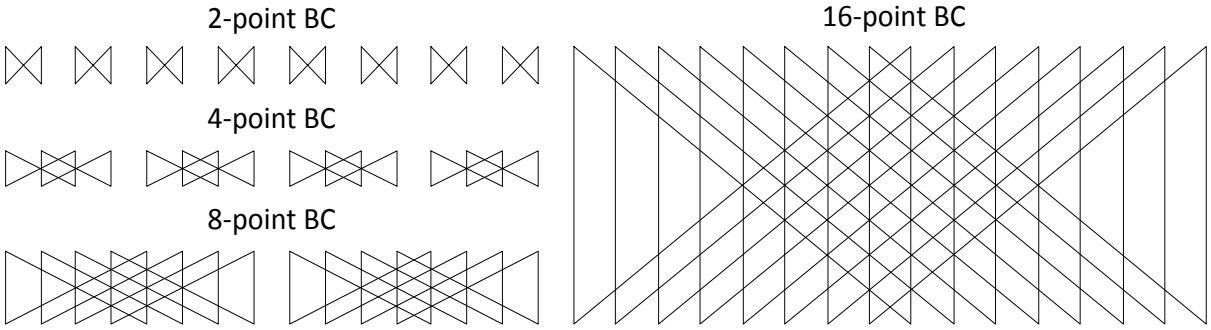

Figure 18 – BC blocks interposition according to the BC size

$$E_k = \frac{x_k + x_{N-k}}{2} \tag{20}$$

$$O_k = \begin{cases} \frac{x_k - x_{N-k}}{2} & \text{, for } k < \frac{N}{2} \\ \frac{x_{N-k} - x_k}{2} & \text{, for } k >= \frac{N}{2} \end{cases} \tag{21}$$

4.1.3.4 Output rotation block

The output rotation block dataflow is similar to the BC block dataflow, where the multiplier $cos\frac{\pi n}{2N}$ is converted in a fixed-point number, converted into additions and sums to perform the multiplication, and the result is right-shifted to compensate the multiplier conversion into fixed-point. Moreover, since the multiplier is dependent on the block size, five multiplications are performed and selected according to the block size, as shown in Figure 19(c).

As mentioned in section 2.1.2.1, the HEVC DCT is a 2D-DCT. However, the real

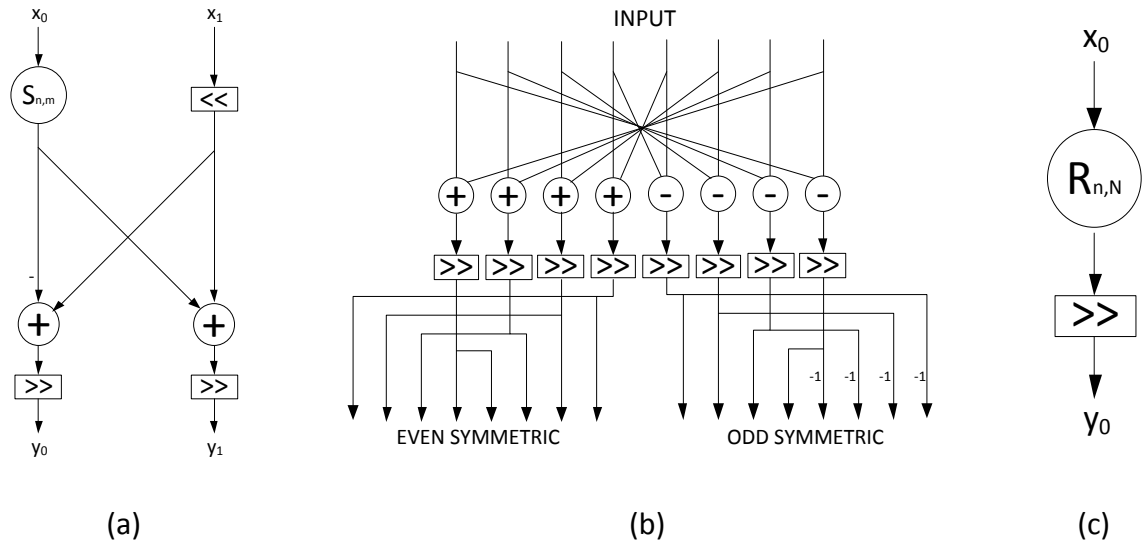

Figure 19 – Proposed DCT sub block data flow. (a) BC block. (b) EOD block. (c) Output rotation block.

challenge in the 2D-DCT implementation is in the 1D-DCT module. Therefore, only the first 1D-DCT is implemented, since the second 1D-DCT has similar cost and performance.

4.2 HEVC 1D-IDCT Hardware Designs

A hardware design implementing the 1D-IDCT algorithm is described in subsection 4.2.1. Such hardware design is used as the base for the implementation of an energy/quality scalable hardware design for the first 1D-IDCT, described in subsection 4.2.2.

4.2.1 Regular 1D-IDCT

The regular 1D-IDCT hardware design is based on the block diagram in Figure 20. The design reuses hardware by calculating the even-rows samples using the immediate smaller 1D-IDCT. In Figure 20, there is one 32-point 1D-IDCT block, one 16-point 1D-IDCT block, two 8-point 1D-IDCT blocks and four 4-point 1D-IDCT blocks. By combining some of these blocks, the design can process 32-point columns, regardless the TB quadtree structure. On the other hand, the design in Figure 20 has blocks not being used for processing the 1D-IDCT at any time. Nonetheless, such blocks consume static and/or dynamic energy. Static energy consumption is unavoidable unless power gating (SHIN et al., 2010) is used. However, the dynamic energy consumption in unused blocks can be suppressed if there is no data toggling in their inputs. Therefore, by inserting clock-gated registers in the blocks inputs, the dynamic energy consumption

can be suppressed in unused blocks, reducing the overall energy consumed by the design. Hence, regardless the combination of blocks, the design can constantly process 32 samples per cycle, while mitigation dynamic energy consumption by disabling data toggling in the unused blocks.

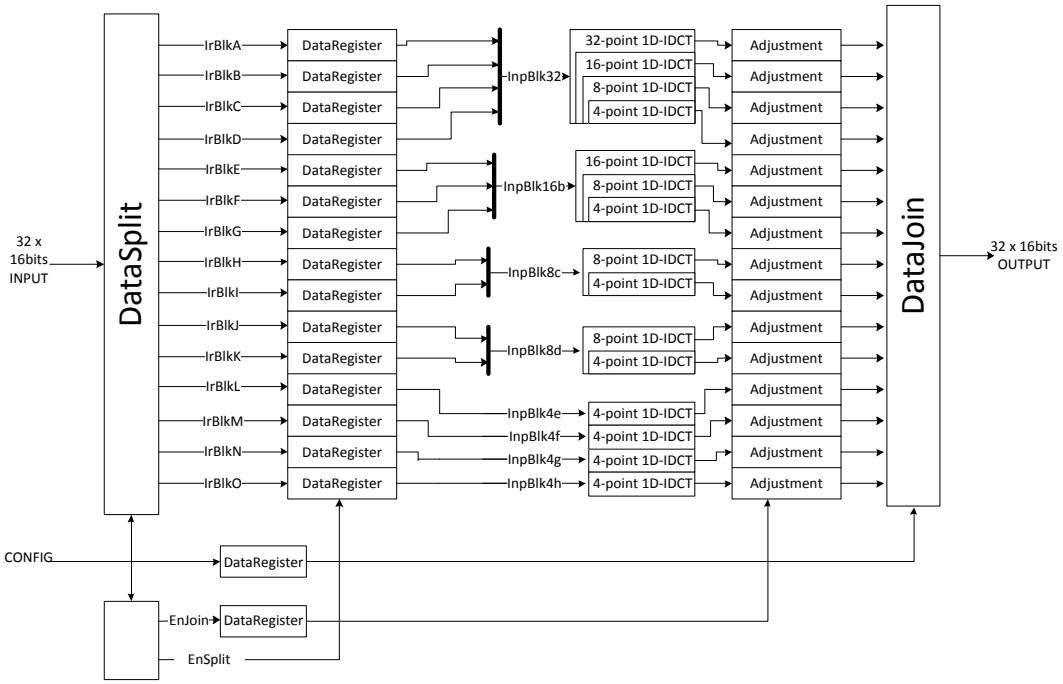

Figure 20 – Regular 1D-IDCT block diagram

The INPUT data is 32 16-bit samples. Along with input data, a control input named CONFIG is used to specify the organization of the column. The CONFIG input is 16-bit wide, grouping eight 2-bit codes. Each 2-bit code identifies the block size for four samples of the input data, coded according to Table 13. The CONFIG input distributes the INPUT data to the appropriate 1D-IDCT blocks and reorganizes the outputs of the 1D-IDCT blocks to the right order.

Table 13 – Encoded TB size pattern

Code	TB size
00	4 x 4
01	8 x 8
10	16 x 16
11	32 x 32

The INPUT data is applied to the DataSplit block. The DataSplit block distributes data between IrBlkA to IrBlkO signals, according to the value of CONFIG input. The IrBlkA is a 256-bit wide signal (sixteen 16-bit samples) which carries odd samples for the 32-point 1D-IDCT. IrBlkB and IrBlkE signals are 128-bit wide (eight 16-bit samples) carrying odd samples for 16-point 1D-IDCTs. The remaining signals are 64-bit wide

(four 16-bit samples), where IrBlkC, IrBlkF, IrBlkH, and IrBlkJ signals carry odd samples for 8-point 1D-IDCTs, and IrBlkD, IrBlkG, IrBlkI, IrBlkK, IrBlkL, IrBlkM, IrBlkN, and IrBlkO transport data for 4-point 1D-IDCTs.This distribution is controlled by the CONFIG signal. At the end of the architecture, the CONFIG signal is responsible for the reorganization of distributed data to maintain the original partitioning structure.

The CONFIG signal is also decoded into two signals:

- *EnSplit*: The EnSplit signal is responsible for the clock gating in DataRegister blocks. The EnSplit signal is applied to the ENABLE input in DataRegister blocks for the regular 1D-IDCT architecture or is combined with the BYPASS signals to define the clock gating in the approximate architecture.

- *EnJoin*: The EnJoin signal bypasses the Adjustment blocks when a 1D-IDCT output is not being used or is used as part of a bigger sized 1D-IDCT calculation.

The outputs of DataRegister blocks related to IrBlkA, IrBlkB, IrBlkC, and IrBlkD are grouped together to form InBlk32 signal, which is the input to the 32-point 1D-IDCT operational unit (OU). Similarly, the output of DataRegister blocks related to IrBlkE, IrBlkF,andIrBlkGare grouped to form IrBlk16b signal, the input to the second 16-point 1D-IDCT OU. The first 16-point 1D-IDCT OU is part of the 32-point 1D-IDCT OU, where the 16-point 1D-IDCT OU processes the even samples for the 32-point 1D-IDCT OU. The output of DataRegister blocks related to IrBlkG and IrBlkH, and IrBlkI and IrBlkJ are grouped to form the IrBlkc8c and IrBlk8d signal, respectively. These signals are inputs to the third and fourth 8-point 1D-IDCT OUs.

As above mentioned, there is a hierarchical OUs usage, where the even samples of an OU are processed by the immediate smaller OU. For instance, the first 16-point 1D-IDCT OU is embedded in the 32-point 1D-IDCT OU. When an embedded OU is processing even samples from its container OU, step 4 in the EOD algorithm, shown in Figure 8, should not be executed. On the other hand, when an embedded OU is assigned to process part of the input data as a whole block, the step 4 in the EOD algorithm must be executed. This step is a combination of a sum and a shift, and it is implemented in Adjustments blocks. Dynamic energy consumption can be saved by disabling the Adjustment blocks when they are not being used. The energy saving can be achieved by an Operand Isolation (CORREALE, 1995) technique, controlled by the EnJoin signal. At last, the outputs of the Adjustment blocks are combined in the DataJoin block, also controlled by the CONFIG input. The EnSplit and EnJoin signals are desynchronized, since the EnSplit signal actuate on the DataRegister blocks input data, and the EnJoin signal actuate on the DataRegister blocks output data, lagging one clock cycle from EnSplit actuation. Therefore, these signals must be registered to maintain synchronization.

4.2.2 Energy/quality scalable solution for the first 1D-IDCT

Input data for the first 1D-IDCT come from the inverse quantization module, which usually has many zero RTR values. Columns with up to one non-zero value are very common (CONCEICAO et al., 2015). Such columns are thereafter called almost zeroed columns (AZC). CONCEICAO et al. (2015) showed that the first 4-point 1D-IDCT presented a minimum of 74% AZC occurrence.

Let $F = [A, 0, 0, 0]^T$ be a 4-point AZC, where A can be any integer number, including zero. The first 1D-IDCT transform, given by (22) in matrix notation, is applied to column F, in (22). The result in (23) is prior to the bitlength adjustment (step 4 in Figure 8). After the bitlength adjustment for the first 1D-IDCT in (24), the final first 4-point 1D-IDCT is given by (25).

$$f' = C_4^T \times F \tag{22}$$

$$f' = \begin{vmatrix} 64 & 83 & 64 & 36 \\ 64 & 36 & -64 & -83 \\ 64 & -36 & -64 & 83 \\ 64 & -83 & 64 & -36 \end{vmatrix} \times \begin{vmatrix} A \\ 0 \\ 0 \\ 0 \end{vmatrix} = \begin{vmatrix} 64A \\ 64A \\ 64A \\ 64A \end{vmatrix} \tag{23}$$

$$f_1 = (f' + 64) >> 7 \tag{24}$$

$$f' = \begin{vmatrix} \lfloor (A+1)/2 \rfloor \\ \lfloor (A+1)/2 \rfloor \\ \lfloor (A+1)/2 \rfloor \\ \lfloor (A+1)/2 \rfloor \end{vmatrix} \tag{25}$$

The regular 4-point 1D-IDCT calculation in (23) requires 16 multiplications and 12 sums. However, when the input data is AZC-compliant data the matrix is not required, since the final result can be achieved by directly applying (25), bypassing the 1D-IDCT. Moreover, AZC-compliant input processing can be further simplified by replacing the rounding to the nearest integer method with the rounding to the nearest integer towards zero method, as in (23). This replacement inserts some imprecision since it suppresses the offset addition, which affects only the least-significant bit in the output column. Although CONCEICAO et al. (2015) applied this method only for the first 4-point 1D-IDCT, it can be applied to all transform block sizes without loss of generality.

$$f1 = |A \gg 1, A \gg 1, A \gg 1, A \gg 1|^T \tag{26}$$

The complexity reduction technique based on bypass the 1D-IDCT can be precise, when the bypass occurs only when input data is AZC compliant, or approximate, when

the bypass control is scalable based on an input and the statistics of AZC occurrences.

The disadvantage of the precise version is the long data path in the bypass control, especially, for the 16- and 32-point 1D-IDCT. On the other hand, the approximate approach can insert errors in the 1D-IDCT results, since it enables the bypass based on the statistics, disregarding the current input data AZC compliance. As the approximate approach possibly introduces errors in the 1D-IDCT process, we perform an analysis to find the coding efficiency impact of using this 1D-IDCT approach.

4.2.2.1 Statistical analysis of AZC occurrence in HEVC first 1D-IDCT

The statistical analysis of AZC occurrence in HEVC first 1D-IDCT requires data gathering from a modified version of HM. The HM, version 16.7, code is modified to count the number of AZC-compliant columns processed by first 1D-IDCT, grouped by column index and QP value. The overall number of columns processed by first 1D-IDCT, grouped by column index and QP value, are also accounted.

The data is extracted by the encoding of three HD (BQTerrace, Cactus, and Kimono) – and three UHD 4K video sequences (Beauty, Jockey, and ShakeNDry). The video sequences are encoded using low-delay configuration profile and four QP values (22, 27, 32, and 37). Figure 21 shows a collection of charts representing the AZC occurrence probability in the encoded videos. The AZC occurrence probability for the 32-, 16-, 8-, and 4-point first 1D-IDCT are presented in Figure 21(a), Figure 21(b), Figure 21(c), and Figure 21(d), respectively. The X-axis in the charts of Figure 21 represent the column index for each block size, and the Y-axis represent the AZC occurrence probability. The AZC occurrence probability is calculated by the ratio of the number of AZC compliant columns over the number of columns of the data collected in the modified HM. Moreover, the AZC occurrence probability is grouped by QP value, block size, and the column index.

Smaller IDCT block sizes are more likely to present AZC compliant data than the bigger ones. Since smaller transform matrices are sub sampled versions of bigger transform matrices, data from bigger TBs are more spread than in the smaller TBs. Therefore the AZC occurrence probability varies according to the block size. In Figure 21, the minimum AZC occurrence probability are 88.6%, 30.5%, 15.0%, and 2.1%, for 32-point first 1D-IDCT, 16-point first 1D-IDCT, 8-point first 1D-IDCT, and 4-point first 1D-IDCT, respectively. A correlation between QP value and AZC occurrence probability can be also observed. Higher QP values round up to more zeros in the QTR data than smaller QP values. Therefore, the AZC occurrence probability is enhanced with the increase in QP value. All charts in Figure 21 show that the point in purple lines (representing QP=37) are above the points in the other lines (representing QP=22, QP=27, and QP=32) in the same column index. Furthermore, since the HEVC DCT concentrates information on the top-left corner of its output matrix, the AZC occurrence

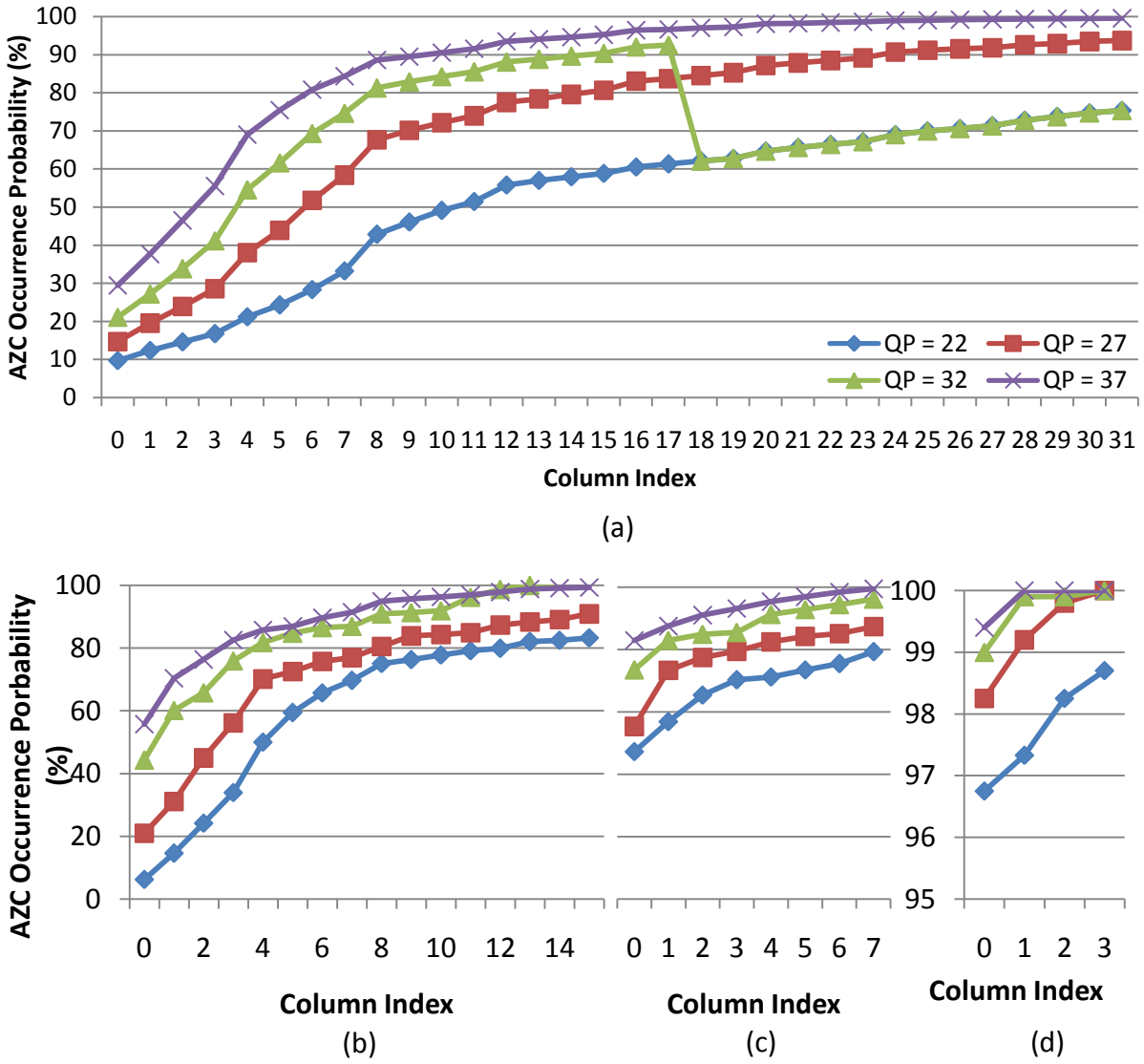

Figure 21 – AZC occurrence probability, grouped by QP and column index. (a) 32-point first 1D-IDCT. (b) 16-point first 1D-IDCT. (c) 8-point first 1D-IDCT. (d) 4-point first 1D-IDCT.

probability is enhanced with the increase in the column index, regardless QP value or block size. Charts in Figure 21 show lines monotonically crescent, except for the green lines (QP=32) in charts (a) and (d).

The data used to plot the charts in Figure 21 compose an offline database that will be used to decide whether a column will be processed by the 1D-IDCT or be bypassed. The bypass control considers an input representing a threshold, i.e. the minimum AZC occurrence probability in which the column must be bypassed, and the offline database data, which is dependent on the QP value, TB size, and column index.

Despite the straightforward applicability of the threshold, the relationship of the threshold and output relevant targets, as power dissipation or impact on coding efficiency, is not that clear. On the other hand, the bypassed columns percentage (BCP)

input has a more direct relationship with the above-mentioned output relevant targets. Nonetheless, BCP is more complex to apply directly in high-throughput energy-aware hardware architectures.

Therefore, the BCP input is converted into a threshold signal which acts on the bypass control, based on the data collected for the statistical analysis of AZC occurrence in HEVC first 1D-IDCT. Such conversion is made by setting the threshold value in the range from 0.00% to 100.0%, with 0.1% step, and calculating the corresponding BCP value, for each QP value used (22, 27, 32, and 37), as shown in Figure 22.

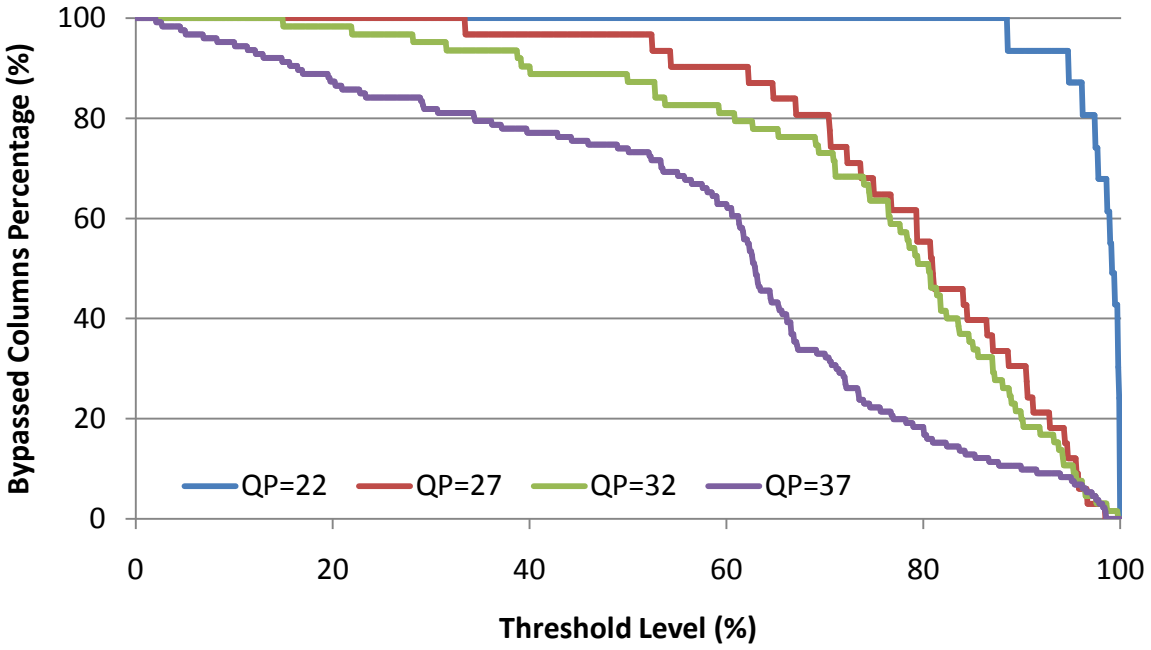

Figure 22 – BCP data as a function of the threshold level and the QP value

The BCP data is plotted as a function of the threshold level in Figure 22, i.e. $BCP(Threshold, QP)$. Nonetheless, the BCP to threshold conversion requires the inversion of such function – $Threshold(BCP, QP)$. This function is inverted by setting the BCP value to the range from 0.00% to 100.0%, with 0.1% step, and searching in the data represented in Figure 22 for the highest threshold value which accomplishes the current BCP value, for each QP value used. Figure 23 show the resulting threshold level as a function of BCP and QP values. Therefore, the BCP input data derives the minimum AZC occurrence probability, i.e. threshold level, which is used by the bypass control.

The four QP values, recommended by the CTC, are used in Figure 23 and Figure 22. However, QP values ranging from 0 to 51 can be set for the video encoding. Moreover, QP values vary on a frame-to-frame basis in Random Access configuration profile and Low Delay configuration profile in HEVC. Therefore, the statistical analysis of AZC occurrence in HEVC first 1D-IDCT must be made to approximate its values to

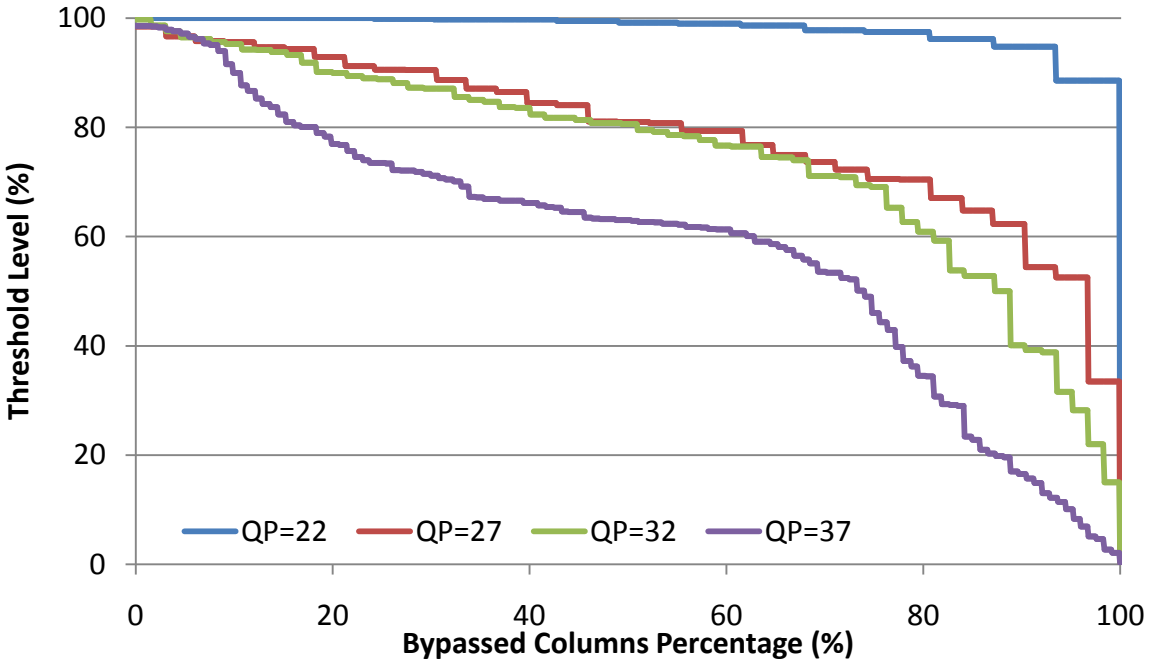

Figure 23 – Threshold level as a function of BCP and QP values

the whole QP range. The approximation consists in framing the input QP value to the QP values used in the statistical analysis, according to (27).

$$QP_{used} = \begin{cases} 22, \text{ if } QP < 25, \\ 27, \text{ if } 25 <= QP < 30, \\ 32, \text{ if } 30 <= QP < 35, \\ 37, \text{ otherwise} \end{cases} \tag{27}$$

4.2.2.2 Impact on Output Encoded Video Size

As previously stated, the approximate approach usage can insert errors in the 1D-IDCT processing. For instance, few columns achieve 100% AZC occurrence probability in Figure 21. Nonetheless, when some column are bypassed using the approximate approach, according to the BCP (Bypassed Columns Percentage) value and the QP value, such column data is not processed by the 1D-IDCT, regardless its AZC compliance. The allowed approximation can affect coding efficiency, e.g. the approximation can lead the encoder to choose sub-optimal prediction modes and/or block sizes, which can increases the bitrate. Therefore, the evaluation of the impact on bitrate according to the BCP value used is required.

The impact analysis was made for three UHD 4K video sequences – HoneyBee, Suzie, and YachtRide – using Low Delay configuration profile, and the CTC recommended QP values. Moreover, eleven BCP values are used: 0%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, and 100%. Table 14 shows the impact on bitrate, measured by the videos BD-Rate, using 0% BCP as the reference.

Table 14 – Impact of bypass usage on coding efficiency

	BD-Rate (%)			
BCP level (%)	HoneyBee	Suzie	YachtRide	Average
10.00	0.0226	0.6826	0.4399	0.3817
20.00	0.0230	0.7471	0.4620	0.4107
30.00	0.0243	0.8648	0.4867	0.4586
40.00	0.0256	0.8938	0.5124	0.4773
50.00	0.0763	1.2451	0.5342	0.6185
60.00	0.0766	1.2447	0.5831	0.8646
70.00	0.1457	2.3693	0.6429	1.0526
80.00	0.1533	2.4088	0.7066	1.0896
90.00	0.2600	2.9921	0.7419	1.3313
100.00	0.2730	3.3759	0.7493	1.4661

Table 14 shows a monotonic growth of the BD-Rate values as the BCP level increases. The BD-Rate increase for Honeybee video can be considered negligible (at most 0.2730%), while the YachtRide video presents low BD-Rate impact (up to 0.7493%). On the other hand, Suzie video presents up to 3.3759% (100% BCP level) BD-Rate increase, the highest bitrate impact. The average BD-Rate impact is considerably low, ranging from 0.3817% to 1.4661%, on 10% BCP level and 100% BCP level, respectively.

The relatively low impact on BD-Rate of the approximate solution (1.47%, on average), along with the perspective of an energy-saving proportional to the BCP value, lead to the development of an energy/quality scalable hardware design, where the trade-off between energy and quality can be adjusted dinamically.

4.2.2.3 Energy/quality scalable 1D-IDCT hardware architecture

The approximate 1D-IDCT hardware architecture is based on Figure 20. The energy/quality scalable (EQS) 1D-IDCT architecture, in Figure 24, adds the blocks and connections in gray, which compose the complexity reduction technique. The complexity reduction technique uses three new inputs – PRECISE, QP, and BCP – and implements a Threshold Generator block, a Bypass Control block, fifteen BypassMux blocks, and five DataRegister blocks. The complexity reduction technique also uses a control signal to keep track of the current column index – ColIndex – and modifies the clock gating signal in the DataRegister ENABLE inputs.

The Threshold Generator block implements a look-up Table (LUT), to provide the appropriate Threshold level signal, according to the QP input and BCP input values. Moreover, the PRECISE input in the Threshold Generator block forces the threshold value to 110%, which will ensure all of the Bypass Control block outputs to be disabled. The Bypass Control implements four LUTs, one for each block size, to select

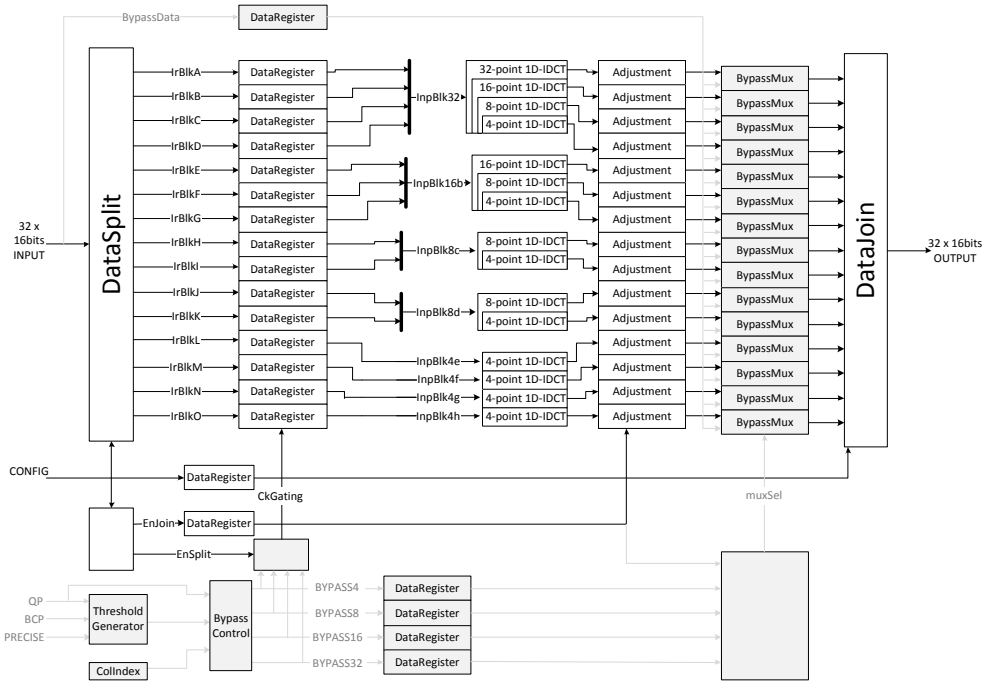

Figure 24 – Bypass controlled approximated 1D-IDCT architecture

the current AZC compliance occurrence rate. Such occurrence rates are compared to the Threshold level to decide which block sizes will be bypassed. The outputs of the Bypass Control block – BYPASS4, BYPASS8, BYPASS16, and BYPASS32 – are set when the respective LUT output is bigger than Threshold level signal.

The BypassMux blocks select between the Adjustment block outputs and the registered first sample relative to each block, the registered BypassData signal, which will be right shifted by one bit and replicated throughout the block output. The BypassMux blocks are controlled by the muxSel signal which combines the registered EnJoin signal with the registered Bypass Control outputs. The first samples of each block, EnJoin signal and Bypass Control outputs are registered to maintain data temporal consistency.

This subsection presented an approximate quality/energy scalable version of the first 1D-IDCT suiTable to power reduction during the RDO evaluation. The quality/energy scalability is provided by a bypass control and its auxiliary blocks. Nonetheless, the hardware design can be used as the second 1D-IDCT or as the first 1D-IDCT used in the actual video coding. This can be accomplished by introducing the PRECISE input, which disables the BypassControl module.

4.3 Project Space Exploration of Quantization Architectures

In this section, many techniques are used to search for the best trade-off between coding efficiency and energy efficiency for the direct and inverse quantization hardware

design. The developed architectures of direct and inverse quantization were designed combining flat-quantization (Flat) or frequency-dependent quantization (FDQ) support and using generic multiplication (GM) or multiple-constant multiplication (MCM). Moreover an integrated direct/inverse quantization was developed aiming throughput increase by reducing the number of operations required to perform the direct- and inverse-quantization.

Regardless the design particularities, both direct and inverse quantization require an input data (transformed residual for direct quantization, and quantized transformed residual for inverse quantization), QP value, and the TB size. Additionally, direct quantization requires an input which indicates if the current encoded slice is intra-coded only (SLICE_I). The FDQ supporting architectures also requires the position of the sample in the TB, to select the proper QStep.

The 32 sample per cycle approach is also used for all direct and inverse quantization designs. Thus, all designs use 32 single-sample cores (SSC) of direct or inverse quantization. Some optimizations can be made, taking advantage of redundancies in the setup of neighbor SSCs. For instance, QP value is constant throughout the TB quadtree root-node. Therefore, regardless the TB partitioning, QP value will be the same for SSCs. Furthermore, (8) and (13) use $\lfloor QP/6 \rfloor$ and $QP\%6$, i.e. the quotient and the remainder of the division of QP by six. Therefore, QP is divided by six in advance and its results are distributed to the 32 SSC instances. Although SLICE_I is also constant throughout the TB, this input is used in (9), combined with qBits in (10), which depends on TB size.

Since the minimum TB size is 4×4, the direct quantization SSCs (DQ-SSC) can be grouped four by four, then the previously calculated qBits and offset signals can be distributed to the grouped SSCs. Inverse quantization SSCs (IQ-SSC) also depend on the results from the division of QP by six, and the TB size. The quotient and remainder of the division of QP by six is distributed for the 32 IQ-SSC instances, whereas offset and the number of bits to shift can be calculated in advanced and distributed to four neighbors SSCs, as occurs in the DQ-SSC. Figure 25 shows a generic parallel quantization component block diagrams. The “Local” blocks, detailed in Figure 26, calculate offset and number of bits to shift, based on TB size, quotient of division of QP by six, SLICE_I (for direct quantization) and row number (for FDQ versions). The “SSC” blocks group four SSC instances which use common parameter calculated by correspondent “Local” block and perform the direct or inverse transform, whether SSC instances are DQ-SSCs or IQ-SSCs, respectively.

TB size input is encoded in the CONFIG input of Figure 25. The CONFIG input is a concatenation of eight two-bit signals, which encode the TB size for each Local block, according to Table 13.

The following subsections describe the specific differences among the developed

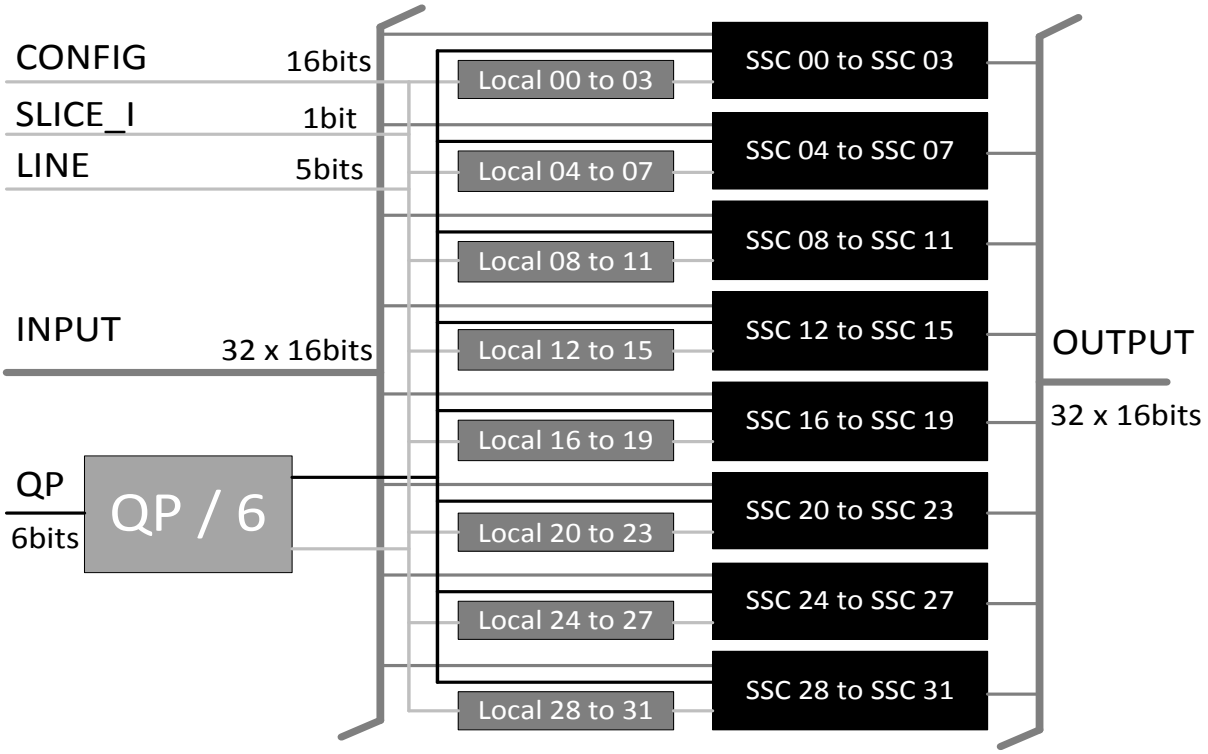

Figure 25 – Top-level block diagram of generic parallel quantization component

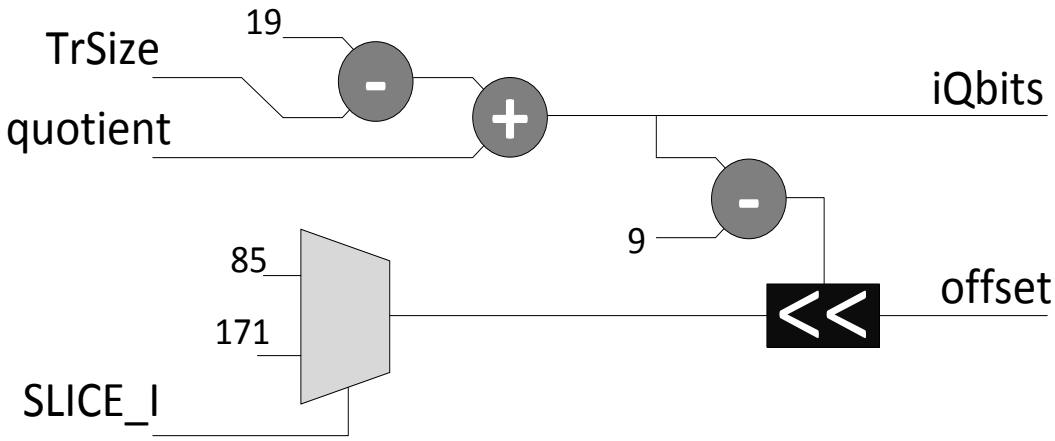

Figure 26 – Local block diagram

SSC designs.

4.3.1 Direct flat-quantization using multiple-constant multiplication

Figure 27 shows the block diagram of the DQ-SSC with flat-quantization using MCM. The input data is the transformed residue (TR), the remainder signal is provided by the $QP/6$ block in Figure 25, offset and iQbits signals come from the respective "Local" block in Figure 25.

The transformed residual has its sign and absolute value extracted: sign(TR) and |TR|. The absolute TR value is multiplied by a constant in the MCM block, according to

the remainder of QP division by six. The multiplication result is added to the offset and shift to the right by “iQbits” bits. At last, the sign(TR) signal is used to apply the sign of the input data TR into the output QTR data.

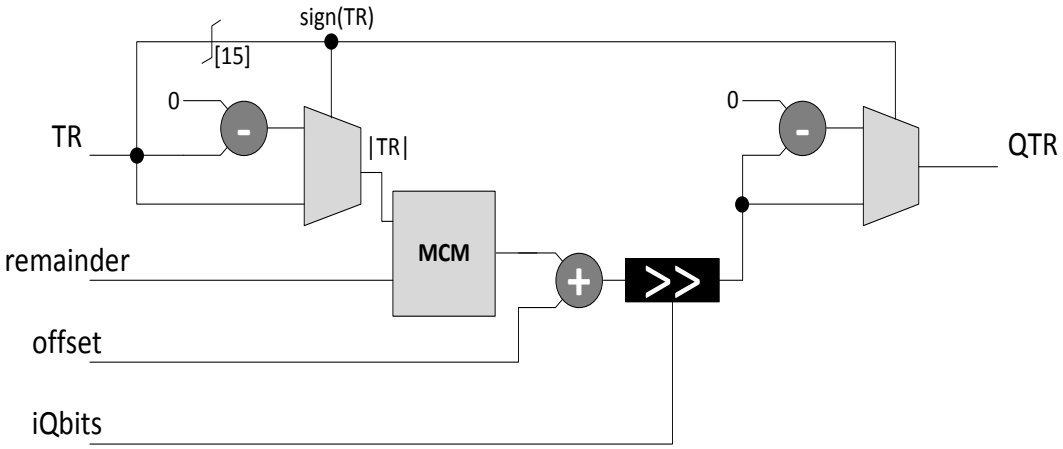

Figure 27 – DQ-SSC with flat-quantization using MCM

Figure 28 details the MCM block used to perform the multiplication by the six constants of $G(QP\%6)$. The multiplications are performed using additions and shifts, reusing intermediate results and balancing out the additions whenever it is possible. At the end, one of the six product results is selected based on the remainder input.

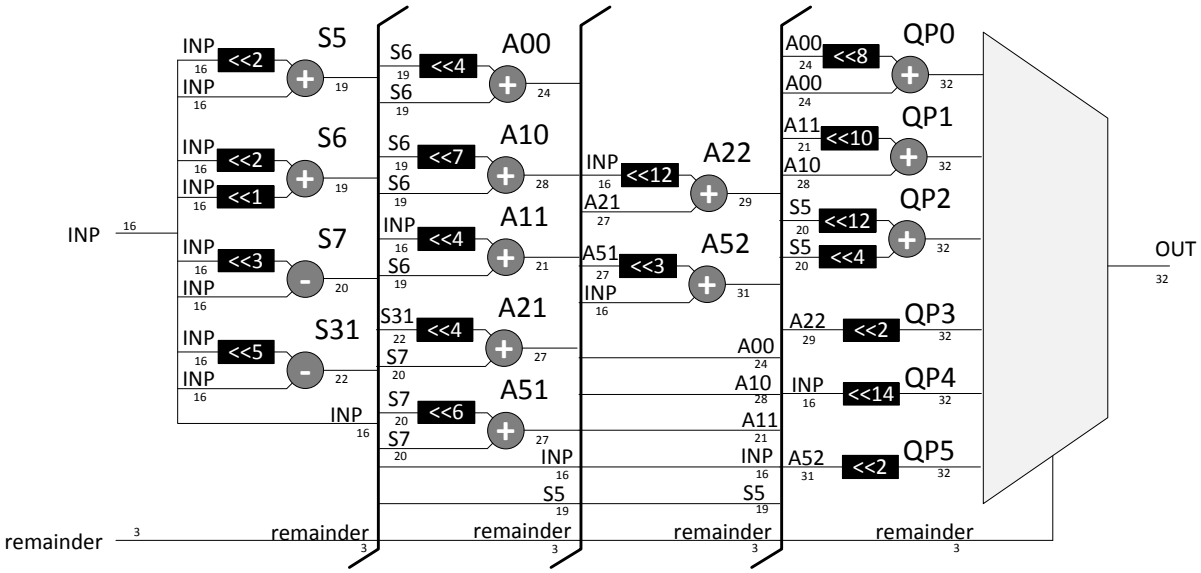

Figure 28 – Block diagram of MCM block for DQ-SSC with flat quantization

The additions distribution aim to maximize reusing operations and balancing the operations to minimize the propagation delay. Table 15 shows the values in $G(QP\%6)$, in decimal, and binary.

The reuse of arithmetic operations can be illustrated with the description of the multiplication by 26214. The multiplication (28) can be performed by using shifted additions

Table 15 – $G(QP\%6)$ constants to be implemented in MCM

QP%6	Decimal	Binary
0	26214	0110.0110.0110.0110
1	23302	0101.1011.0000.0110
2	20560	0101.0000.0101.0000
3	18396	0100.0111.1101.1100
4	16384	0100.0000.0000.0000
5	14564	0011.1000.1110.0100

whenever there is a “1” bit, as in (29), where $<<$ represent left shifts. Expressions in (30) and (31) reduce the number of adders required by the multiplication, by grouping additions with similar shifts in previous expressions, (29) and (30), respectively. Therefore, the multiplying by 26214 can be performed using only three additions (31).

The other coefficients are processed similarly. Moreover, whenever is possible, intermediate operations are reused in other constants calculation to save adders.

$$y = 26214 \times x \tag{28}$$

$$y = x << 14 + x << 13 + x << 10 + x << 9 + x << 6 + x << 5 + x << 2 + x << 1 \tag{29}$$

$$y = (x << 2 + x << 1) \times (1 << 12 + 1 << 8 + 1 << 4 + 1) \tag{30}$$

$$y = (x << 2 + x << 1) \times (1 << 4 + 1) \times (1 << 8 + 1) \tag{31}$$

4.3.2 Direct flat-quantization using generic multiplication

Figure 29 shows the block diagram of the DQ-SSC with flat-quantization using a generic multiplier and a ROM providing the constants that will be multiplied by the input absolute value.

The difference between Figure 27 and Figure 29 is the replacement of the MCM block by the $G(QP\%6)$ ROM block and a generic multiplier. The $G(QP\%6)$ ROM block is composed of a 6-word 16-bit LUT.

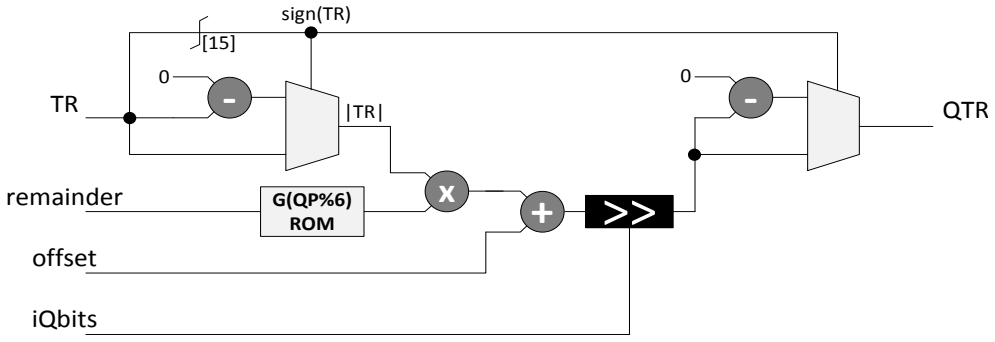

Figure 29 – DQ-SSC with flat-quantization and generic multiplier usage

4.3.3 Direct quantization with FDQ support

Figure 30 shows the block diagram for a DQ-SSC with FDQ support using MCM.

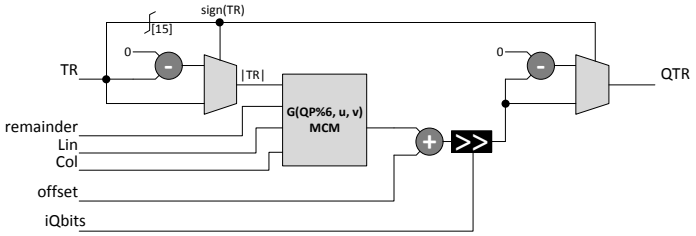

Figure 30 – DQ-SSC with FDQ support using generic multiplier

As mentioned in section 2.1.2.3, FDQ usage increases the number of pre calculated QSteps from the six values in $G(QP\%6)$ to 384 in $G(QP\%6, u, v)$. Therefore, when using MCM, the DQ-SSC with FDQ support uses an MCM block with two additional inputs: the normalized Lin/Col inputs (Figure 30). The FDQ-supporting MCM block diagram is a much bigger version of the MCM block for flat quantization in Figure 26. The FDQ-supporting MCM block uses 207 arithmetic operations, instead of the 14 arithmetic operations in the flat-quantization MCM block. Moreover, while flat-quantization MCM has one multiplexer to select one among six multiplication results, the FDQ-supporting MCM block has a two-step selection structure, shown in Figure 31. In the first step, it uses the remainder input, the normalized Lin input, and the normalized Col input to select an index signal. The index signal previously defined selects one among 213 unique FDQ constant multiplication results.

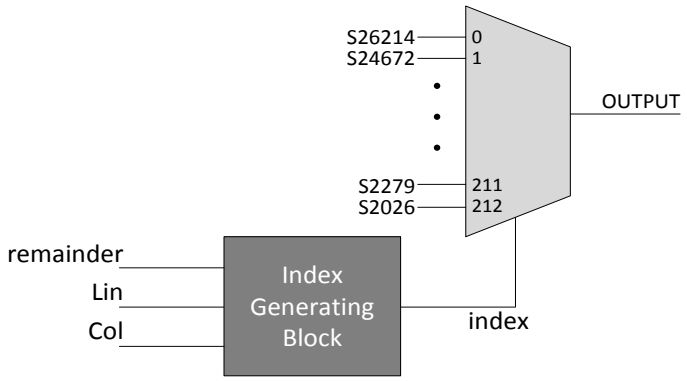

Figure 31 – Two-step selection structure for the FDQ-supporting MCM block

The difference between Figure 29 and Figure 30 is in the ROM. The flat-quantization DQ-SSQ in Figure 29 implements a 6-word 16-bits QStep values, where the DQ-SSC with FDQ support implements a 384-word 16-bits QStep values. Such 384-words are the six 8×8 quantization matrices, described in section 2.1.2.3. Furthermore, the ROM uses the remainder and the position (inputs Lin and Col) of the TR in the TB to select the FDQ constant.

Althought the line indexes and column indexes can range from 0 to 31, Lin and Col inputs must be constrained to the 0 to 7 range, with the coefficient replication described in section 2.1.2.3. Figure 32 shows the Lin/Col value normalization diagram block, based on the TB size. According to section 2.1.2.3, FDQ is not applicable to 4×4 TBs. Therefore, when TB size is 4×4, the normalized Lin/Col is always set to zero. The twice-replicated 16×16 QMs is applied by discarding the least-significant bit in the Line/Column for the 16×16 TB. Similarly, the four times replicated 32×32 QMs is accomplished by discarding two least-significant bits in the Line/Column for the 32×32 TB. This block is incorporated into the Local block in Figure 26 and distributed to the four grouped SSCs related to it.

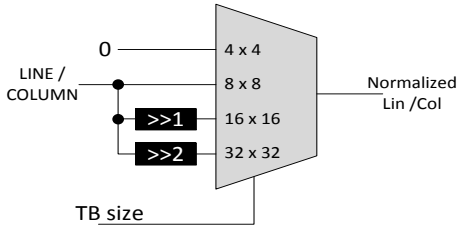

Figure 32 – Lin/Col value normalization

4.3.4 Generic inverse quantization single-sample core

Figure 33 shows the block diagram of the IQ-SSC with FDQ support using a generic multiplier. The inverse quantization uses the quotient and the remainder of the division of QP by six, the offset and shift inputs are derived from the block size in the correspondent Local block. The inverse quantization main input is the quantized-transformed residual (QTR) and its output is the reconstructed-transformed residual (RTR). The inverse quantization is processed by multiplying QTR by a constant in $h(QP\%6, u, v)$, shifted to the left by as many bits as the quotient of the division of QP by six. Then, the shifted product is added to the offset input and shifted to the right, according to the shift input.

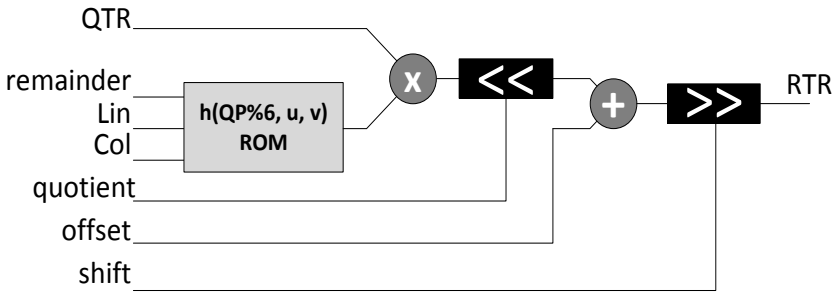

Figure 33 – IQ-SSC with FDQ support using generic multiplier

The IQ-SSC is also implemented in all variations described in DQ-SSC: IQ-SSC with FDQ support using a generic multiplier, IQ-SSC with FDQ support using MCM,

flat IQ-SSC using a generic multiplier, and flat IQ-SSC using MCM. As it occurs in the DQ-SSC, the FDQ support on IQ-SSC requires the additional Lin and Col inputs, which are the normalized position of the QTR sample in the TB block. Furthermore, the multiplier and ROM blocks are replaced by an MCM block in IQ-SSCs using MCM, as it is in DQ-SSCs.

4.4 Integrated Direct/inverse Quantization hardware design

Hardware implementations of the direct quantization and inverse quantization, such as DIAS; ROMA; SOUSA (2015) and BRAATZ et al. (2017), are based on equations (8) and (13), respectively. Despite the optimizations applied to (8) and (13) aiming hardware friendliness, such equations still present a high propagation time, since there are sequences of arithmetic operations processing wide numbers. Replacing generic multiplication by MCM, can reduce hardware area requirements and shorten propagation delays. Nonetheless, a combination of direct and inverse architectures present 13 levels of data dependency, in the best-case scenario. Such data dependency depth reduces the available throughput in a complete RCL hardware solution. Although the usage of pipeline registers (DIAS; ROMA; SOUSA, 2015) reduces the propagation delay, it also increases latency and energy consumption. RCL components must present the lowest propagation delay and the lowest latency possible, to speed up the RDO evaluation process.

In this work, a new approach proposing a high-throughput integrated direct/inverse quantization (IDIQ) design is developed. The IDIQ module is designed to speed up the RDO evaluation process. However, the IDIQ module does not provide the intermediate quantized-transformed residual (QTR), as shown in Figure 34, therefore it cannot be used in the final video encoding. Thus, the IDIQ module is designed to be inserted in parallel with the regular direct and inverse quantization, as shown in Figure 34. When the TR data come from the RDO evaluation process, they are processed by the IDIQ module and when TR data come for the final video encoding, they are processed by the regular direct and inverse quantization. Nonetheless, the RDO provided TR data use the reconstructed transformed residual (RTR) as an approximate bitstream to evaluate the block bitrate.

The IDIQ architecture presents higher throughput when compared to the cascaded regular direct quantization and inverse quantization. Therefore, IDIQ is used to process TBs during the RDO evaluation of different CM. At last, the chosen CM is processed by the direct and the inverse quantization blocks. Then, QTR data is processed by the entropy coder, and RTR data is used in the block reconstruction for reference frame composition.

The IDIQ increases throughput by reducing the number of arithmetical operations

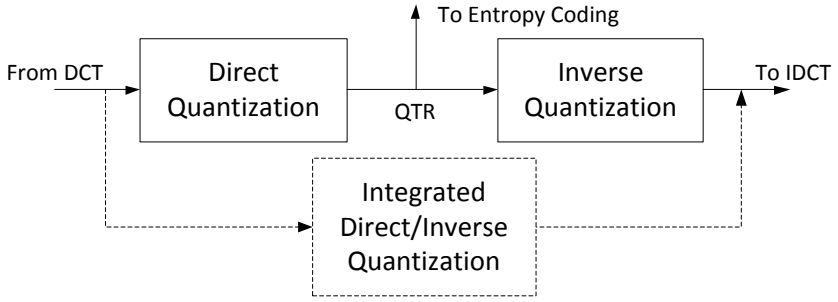

Figure 34 – Integrated Direct/Inverse Quantization placement

required to obtain RTR. Conceptually, the direct quantization and inverse quantization are described by (32) and (33),respectively, where QStep is given by (7). The operator $\lfloor . \rfloor$ in (32) denotes a rounding towards zero operation. Rounding the division towards zero in (32) implies in discarding the remainder of the integer division. In addition, the multiplication in (33) provides an approximate value of TR, where the difference between RTR and TR is the remainder of the division in (32). Thus, RTR is obtained in (34) as a function of TR and QStep.

$$QTR = \left\lfloor \frac{TR}{QStep} \right\rfloor \tag{32}$$

$$RTR = QTR \times QStep \tag{33}$$

$$RTR = TR - (TR mod QStep) \tag{34}$$

Using (34) to calculate RTR reduces the number of arithmetic operations and data dependency depth to two: one modulus operation followed by one subtraction. Nonetheless, the modulus operation is a very costly operation regarding computational effort. Usually, the modulus operation is performed by calculating divisions and multiplications (BARRET, 1987) or using iterative processes (MONTGOMERY, 1985). Both solutions present high propagation delay and/or high latency.

Since there is a limited number of QStep values, it is possible to calculate in advance the results from TR mod QStep and implement such operation as a collection of look-up Tables (LUTs). The QStep possible values are calculated using (13), which is dependent on the QP value and offset and shift signal. Such signals depend on the TB size. Therefore, the QStep possible values are determined with Z value fixed in one, the QP value ranging from 0 to 51, and TB size varying among 4, 8, 16, and 32 values. Overall, there are 208 possible evaluations. The W' value in these evaluations are the QStep constants, composed of 65 unique constants ranging from 3 to 7296. The complete set of constant is available in Table 47, in appendix D. Therefore, the modulus operation is implemented as a LUT using a combination of the dividend and divisor used as the LUT inputs.

Even using LUTs to simplify the modulus calculation, such operation can be area and power inefficient. The modulus operation in (34) uses a 15-bit dividend and a 13-bit divider, in the worst-case scenario. Therefore, the LUT performing the modulus operation would have to select one among 228 15-bit words, requiring a 4.03×10^9 -bit ROM and 28-bit control input. Alternatively, the LUTs can be simplified by splitting the dividend into three 5-bit blocks. Thus, the ROM size in the LUT would range from 1920 bits (27 15-bit words) 3.96×10^6 bits (218 15-bit words), according to the length of the QStep value. The overall results can be obtained by adding the results from the three LUTs, constrained to the range between 0 and $QStep - 1$.

Figure 35 shows the IDIQ Single-Sample Core (SSC) block diagram. In the IDIQ SSC block diagram, the sign and the absolute value of TR input are extracted. Simultaneously, the QStep block defines the QStep value, based on the QP value and the TB size. The QStep value is used as the divisor in all of the MOD blocks. The TB absolute value is split into three 5-bits subsets: the subset using bits 14 to 10 is the dividend for the modulus operation performed in the MOD_C block, the subset with bits 9 to 5 is the dividend for the MOD_B block, and the subset using bits 4 to 0 is the MOD_A block. The outputs of MOD_x blocks are added up and subtracted from the absolute TR value. After each addition, the modulus is extracted based on the QStep value, to maintain the sum of the outputs of the MOD_x block smaller than the QStep value. The RTR output value is obtained by applying the sign extracted from TR into the subtraction results.

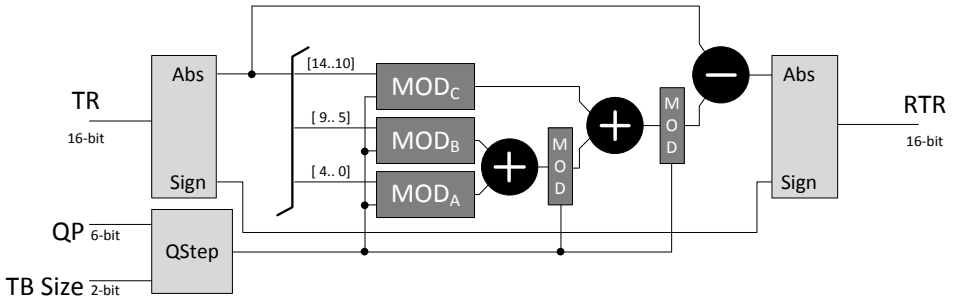

Figure 35 – IDIQ SSC block diagram

A parallel integrated direct/inverse quantization (PIDIQ) hardware architecture, shown in Figure 36, allows the processing of 32 samples per cycle. Furthermore, the parallelization can take advantage of redundancies in the characteristics of neighbor samples. For instance, the QP value is always the same for the 32 samples. Moreover, the TB size is the same at least for four consecutive samples. Therefore, the QStep block can be extracted from the IDIQ block. Thus, there are eight QStep blocks in the PIDIQ architecture, where each of them provides the QStep value for four IDIQ blocks.

In conclusion, all the hardware designs proposed in this chapter can be integrated to form a complete HEVC RCL hardware architecture. Such hardware architecture also can instantiate in parallel the IDIQ hardware design (from section 4.4) and the DQ-SSC

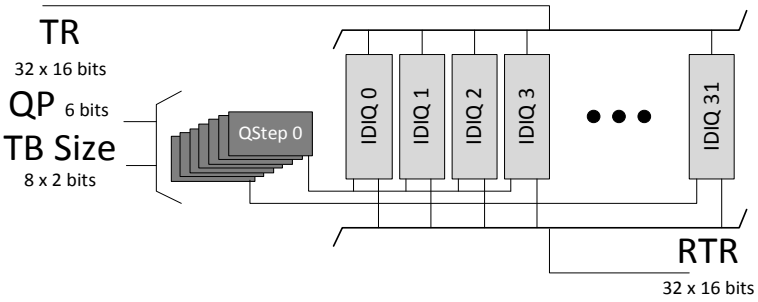

Figure 36 – PIDIQ architecture block diagram

and IQ-SSC (from section 4.3) hardware architectures, to improve throughput during RDO evaluation and ensuring an HEVC compliant bitstream in the final step of the encoding. The EQS IDCT-1D hardware design, in section 4.2, can be dynamically configured to focus on throughput improvement or energy saving.

5 SYNTHESIS RESULTS AND COMPARISONS

The developed hardware architectures were described in VHDL at Register-Transfer Level (RTL). Next, the hardware architectures were validated using the ModelSim 10.1d (MODELSIM, 2018) software, using a set of 32,000,000 real input/output data of each RCL component, gathered from the HM (FLYNN et al., 2011), to validate the hardware designs. The validations consist in using the input data gathered from HM as the input of the respective hardware design and comparing the hardware design output with the respective output data gathered from HM.

Once described in VHDL and validated, the designs were synthesized using the Cadence Encounter® RTL Compiler (RC) version 11.10 (CADENCE, 2018) and the 45-nm Nangate Open Cell Library (NANGATE, 2018) standard cell library, characterized under the PVT (Process Voltage-Temperature) of $1.1V$ and $25°C$. Since the related works may not use the same tools for its synthesis, the occupied area of the hardware designs is presented in NAND2X1 gates area, for comparison purposes. The 45-nm Nangate OpenCell Library NAND2X1 gate, the two-input NAND gate with a fan out of one, occupies an area of $0.798\mu m^2$.

Moreover, during the synthesis process, the power dissipation is estimated. The RC software uses one of the three methods to estimate net switching: 1) using predefined input data toggling and estimating the power dissipation by analyzing the switching propagation based on the hardware RTL description; 2) using statistical analysis based on VCD (Value Change Dump) or TCF (Toggle Count File) switching files; 3) using statistical analysis based on VCD (Value Change Dump) or TCF (Toggle Count File) switching files generated by simulation of the hardware netlist, provided by RC synthesis tool. The results of power dissipation for the developed hardware architectures were generated based on the second net switching method, stimulated by real input data gathered from the HM.

5.1 FFT-inspired HEVC 1D-DCT synthesis results

The FFT-inspired HEVC 1D-DCT hardware design is described according to subsection 4.1. The presented FFT-inspired HEVC 1D-DCT hardware architecture results are listed in Table 16, using different targets. Along with the maximum frequency target, four more syntheses were performed targeting the minimum frequency for real-time video encoding of HD 1080p at 30fps and UHD 4K at 60 fps able to evaluate up to 101 CM, which is the average number of CM processed by the DCT, as described in section 3.2, and single CM, for comparison purpose, since most of the related works focus on single CM targeting. Table 16 presents the results of area (number of gates), frequency (MHz) and dissipated power (mW) for the five targets above mentioned.

Table 16 – Synthesis Results for FFT-inspired HEVC 1D-DCT

Target	Area (gates)	Frequency (MHz)	Power (mW)
Maximum	87,970	2,538.07	12.33
HD 1080p @30fps – 1 mode	87,386	2.92	0.02
HD 1080p @30fps – 101 modes	87,519	294.52	1.54
UHD 4K @60fps – 1 mode	87,423	23.33	0.13
UHD 4K @60fps – 101 modes	87,745	2,356.13	11.61

At the maximum frequency target, the synthesis resulted in an area equivalent to 87,970 gates with a power dissipation of 12.33 mW at 2.54 GHz. The real-time encoding of UHD 4K videos at 60fps with 101 CM requires 2.36 GHz, resulting in an area of 87,845 gates and a power dissipation of 11.61 mW is related to such target. The real-time encoding of HD 1080p videos at 30fps with 101 CM requires 294.52 MHz, such target results in a synthesis with 87,519 gates and dissipates 1.54 mW. Regarding one mode encoding, the real-time encoding of UHD 4K videos at 60fps results in 87,423 gates of equivalent area and 0.13 mW of power dissipation, when operating at 23.33 MHz. The one-mode real-time encoding of HD 1080p videos at 30fps requires an operational frequency of 2.92 MHz and an equivalent area of 87,386 gates while dissipating 0.019 mW.

Table 17 shows a comparison of the five target synthesis of the developed FFT-inspired 1D-DCT and related works. The Table confronts the technologic node, equivalent area in gates, operating/maximum frequency, power dissipation, and the energy consumed per sample processed (EPS – Energy Per Sample). The technology node column expresses the channel width of the cells used in the ASIC synthesis, or it informs the use of FPGA (Field Programmable Gate Array) devices. The area column shows the resulting area of the synthesis, counted as the number of NAND2X1 gates for ASIC synthesis or number of LUTs (look-up Tables) for FPGA synthesis. The frequency column shows the operating frequency (MHz) of the listed works, although it

is not certain if the declared frequency is the maximum frequency of the hardware architecture or the frequency required to the hardware architecture to achieve the preset target of the related work. The power column shows the dissipated power (mW) of the listed works. The related works present the same uncertainty as in the frequency column in the sense that it is unknown which frequency is used to estimate the power dissipation, the maximum frequency or the frequency required to achieve some target. The EPS column expresses the amount of energy consumed to process one sample. This EPS column value is calculated by the product of the dissipated power and the time required for processing each sample.

Table 17 – Comparison of FFT-inspired HEVC 1D-DCT hardware architecture with related works

	Technology	Area (gates)	Frequency (MHz)	Power (mW)	EPS (pJ)
1D-DCT – Max	45nm	87,970	2,538.07	12.33	0.15
1D-DCT – 1 mode HD	45nm	87,386	2.92	0.02	0.21
1D-DCT – 101 modes HD	45nm	87,519	294.52	1.54	0.16
1D-DCT – 1 mode 4K	45nm	87,423	23.33	0.13	0.17
1D-DCT – 101 modes 4K	45nm	87,745	2,356.13	11.61	1.15
Regular 1D-DCT	45nm	138,849	503.82	609.76	25.82
(RENDA et al., 2017)	90nm	29,200	324.00	6.63	0.64
(JRIDI; MEHER, 2016)	FPGA	1760	421.94	16.71	1.24
(BONATTO et al., 2017)	65nm	124,800	185.50	30.56	24.09
(MASERA; MARTINA; MASERA, 2017)	90nm	74,000	250.00	N/D	N/A
(CHATTERJEE; SARAWADEKAR, 2017)	FPGA	924	183.30	N/D	N/A
(BASIRI; MAHAMMAD, 2017)	45nm	42,578	714.4	6.06	0.27
(GOEBEL et al., 2016)	45nm	97,300	50.00	24.2	15.13

The acronym ‘N/D’ in power column of MASERA; MARTINA; MASERA (2017) and (CHATTERJEE; SARAWADEKAR, 2017) rows means that the authors did not provide such information, while the acronym ‘N/A’ means not applicable. The first five rows in Table 17 show the results of the developed FFT-inspired 1D-DCT hardware architecture, regarding the five targets shown in Table 16. The area of the developed FFT-inspired 1D-DCT hardware architecture is between 87,423 and 87,970 gates. Such area is about three times larger than the area in RENDA et al. (2017), about twice the area in BASIRI; MAHAMMAD (2017), and 1.18 times larger than the area in MASERA; MARTINA; MASERA (2017). The developed FFT-inspired 1D-DCT hardware architecture is also 1.11 times smaller than the area in GOEBEL et al. (2016) and 1.43 times smaller than the area in BONATTO et al. (2017). The FPGA synthesized works (JRIDI; MEHER, 2016) and (CHATTERJEE; SARAWADEKAR, 2017) cannot be compared to the developed FFT-inspired 1D-DCT due to the difference in the synthesis technology.

The maximum frequency of the developed FFT-inspired 1D-DCT is at least 3.55

times higher than the frequencies of the related works (BASIRI; MAHAMMAD, 2017).

The developed FFT-inspired 1D-DCT dissipates 12.33 mW at its maximum frequency, such power is twice the power of RENDA et al. (2017) and 2.5 lower than BONATTO et al. (2017). Nonetheless, the comparison of power dissipation among hardware architectures using different synthesis technologies or operating at different frequencies is uneven. Therefore, the Energy Per Sample (EPS) metric is adopted to simplify and equalize the energetic efficiency comparison. The developed FFT-inspired 1D-DCT consumes between 0.15 pJ per processed sample and 0.21-pJ per processed sample, for maximum frequency and for real time encoding of HD 1080p videos at 30 (single CM) targeting, respectively. The related works consume between 0.27 pJ per processed sample (BASIRI; MAHAMMAD, 2017) and 24.09 pJ per processed sample (BONATTO et al., 2017) per sample. Moreover, a Regular HEVC 1D-DCT hardware architecture, implemented exactly as the algorithm on the HEVC reference software (FLYNN et al., 2011) and synthesized with the same libraries and tools as the FFT-inspired 1D-DCT, is presented for comparison purposes. Since the FFT-inspired 1D-DCT hardware architecture presents an area over 15 times smaller, a maximum operation frequency five times larger, dissipates 50 times less power, and consumes 50 times less energy per sample than the Regular 1D-DCT hardware architecture, it can be stated that the improvement on throughput and energy in the FFT-inspired 1D-DCT hardware architecture is majorly due to the algorithmic approach used, rather than the exploration of characteristics of smaller technological node used in the syntheses.

Thus, Table 17 shows the developed FFT-inspired 1D-DCT hardware architecture with the highest frequency and the lowest EPS value, although the developed FFT-inspired 1D-DCT hardware architecture does not present the lowest power dissipation in Table 17. Such results make sense, once the dissipated power is proportional to the frequency, and the EPS depends on the dissipated power and the data throughput. Despite the increase in power caused by high frequency, the parallelism level of the FFT-inspired 1D-DCT increases its data throughput, and reduces the EPS ratio and consequently increases the energy efficiency of the FFT-inspired 1D-DCT hardware architecture.

Table 18 presents the analysis of the coding efficiency impact considering all the works listed in Table 17. Table 18 shows the number of processed coding modes (#CM) in real time and BD-Rate impact of the throughput limitation of real-time encoding of HD 1080p videos at 30fps and UHD 4K videos at 60fps. The number of processed CM is determined by the ration between the throughput of the work and the number of samples to be processed in a 1-second video in each resolution. The BD-Rate impact of the throughput limitation is estimated and based on those number of CM and the expressions (16) and (17) in section 3.2.

The acronym 'N/D' in power column of MASERA; MARTINA; MASERA (2017) and

CHATTERJEE; SARAWADEKAR (2017) rows means that the authors did not provide such information, while the acronym 'N/A' means not applicable. The first five rows in Table 17 show the results of the developed FFT-inspired HEVC 1D-DCT hardware architecture, regarding the five targets shown in Table 16. The area of the developed FFT-inspired HEVC 1D-DCT hardware architecture is between 87,423 and 87,970 gates. Such area is about three times larger than the area in RENDA et al. (2017), about twice the area in BASIRI; MAHAMMAD (2017), and 1.18 times larger than the area in MASERA; MARTINA; MASERA (2017). The developed FFT-inspired HEVC 1D-DCT hardware architecture is also 1.11 times smaller than the area in GOEBEL et al. (2016) and 1.43 times smaller than the area in (BONATTO et al., 2017). The FPGA synthesized works (JRIDI; MEHER, 2016) and (CHATTERJEE; SARAWADEKAR, 2017) cannot be compared to the developed FFT-inspired HEVC 1D-DCT due to the difference in the synthesis technology.

The maximum frequency of the developed FFT-inspired HEVC 1D-DCT is at least 3.55 times larger than the largest frequency in the related works (BASIRI; MAHAMMAD, 2017).

The power dissipation analysis requires equivalent frequency to ensure a fair comparison. Nonetheless, the developed FFT-inspired HEVC 1D-DCT dissipates 12.33 mW when working at 2.54 GHz. The only related works presenting power dissipation lower than the developed FFT-inspired HEVC 1D-DCT are 6.63 mW (RENDA et al., 2017) while working at 324 MHz and 6.06 mW (BASIRI; MAHAMMAD, 2016) while working at 714.4 MHz. The power dissipation of the developed FFT-inspired HEVC 1D-DCT working on the most similar frequency to RENDA et al. (2017) is 1.54 mW at 294.52 MHz. Regarding the working frequency of BASIRI; MAHAMMAD (2016), there is no equivalent frequency among the five targets in the developed FFT-inspired HEVC 1D-DCT. However, the power/frequency ratio observed among the five targets in the developed FFT-inspired HEVC 1D-DCT suggests a power dissipation of about 3.47 mW. Therefore, the developed FFT-inspired HEVC 1D-DCT is more energy-efficient than all of the related works.

Hence, the developed FFT-inspired HEVC 1D-DCT hardware architecture show the lowest power when compared with related works, when operating on frequencies similar to the related works. The developed FFT-inspired HEVC 1D-DCT hardware architecture also achieves the largest frequency among all works listed in Table 17.

Table 18 presents the impact analysis of all the works listed in Table 17. Table 18 shows the processing rate and BD-Rate impact of the throughput limitation of real-time encoding of HD 1080p videos at 30fps and UHD 4K videos at 60fps. The processing rate is the ratio of the hardware architecture throughput and the number of samples per second of each video resolution. The BD-Rate impact of the throughput limitation is estimated and based on those processing rates and the expressions (16) and (17)

in section 3.2.

Table 18 – BD-Rate impact of throughput limitation of works listed in Table 17

	HD 1080p @ 30fps		UHD 4K @ 60fps	
	Proc. Rate	BD-Rate* (%)	Proc. Rate	BD-Rate* (%)
1D-DCT – Max	870.39	0.00	108.8	0.00
(RENDA et al., 2017)	111.11	0.00	13.89	19.08
(JRIDI; MEHER, 2016)	144.70	0.00	18.09	13.07
(BONATTO et al., 2017)	13.60	19.58	1.70	57.14
(MASERA; MARTINA; MASERA, 2017)	85.73	0.03	10.72	25.38
(CHATTERJEE; SARAWADEKAR, 2017)	500.69	0.00	15.64	15.27
(BASIRI; MAHAMMAD, 2017)	244.99	0.00	30.62	4.23
(GOEBEL et al., 2016)	17.14	13.85	2.14	52.60

* estimated based on section 3.2

Table 18 shows the number of CM while encoding real-time HD 1080p videos at 30fps being 870.39 for the proposed FFT-inspired 1D-DCT hardware architecture operating in maximum frequency and a number of CM between 13.60 and 500.69 of the related works. Hence, the BD-Rate impact of the throughput limitation is 0.00% for the proposed FFT-inspired 1D-DCT hardware architecture and between 0.00% and 19.58% for the related works. Regarding the results for UHD 4K videos at 60fps, the proposed FFT-inspired 1D-DCT hardware architecture presents a number of CM of 108.8 and a BD-Rate impact of 0.00%, whereas the related works present number of CM ranging from 1.7 and 30.62, with BD-Rate impact between 4.23% and 57.14%.

In conclusion, the developed FFT-inspired 1D-DCT hardware architecture is the only architecture among the works listed in the Table 17 which allows the implementation of the HEVC 1D-DCT with zero impact on the BD-Rate of the encoded video.

5.2 HEVC 1D-IDCT synthesis results

The developed regular 1D-IDCT and the developed EQS 1D-IDCT hardware architectures results are listed in Table 19, using different targets. Moreover, Table 18 shows the power results of the EQS 1D-IDCT hardware architecture using several BCP (Bypass Column Percentage) input values. Table 19 shows the results of equivalent area, the maximum frequency, the power dissipated, and the BD-Rate impact.

Table 19 shows an equivalent area of 94,366 and 165,141 gates for the regular 1D-IDCT and the EQS 1D-IDCT, respectively, where the area difference is due to the bypass control modules. The maximum operating frequency of the regular 1D-IDCT and the EQS 1D-IDCT are 737.46 MHz and 720.46 MHZ, respectively. The concatenated Threshold Generator and Bypass Control blocks in Figure 24 increase the propagation delay in the EQS 1D-IDCT when compared to the regular 1D-IDCT, therefore the fre-

Table 19 – Synthesis Results for Regular and EQS 1D-IDCT

Hardware Architecture	BCP (%)	Area (gates)	Frequency (MHz)	Power (mW)	BD-Rate (%)
Regular 1D-IDCT	N/A	94,366	737.46	15.28	N/A
EQS 1D-IDCT	0	165,141	720.46	16.84	0.00
	10			16.39	0.38
	20			16.18	0.41
	30			15.58	0.46
	40			15.80	0.48
	50			15.34	0.62
	60			15.07	0.63
	70			14.31	1.05
	80			13.38	1.09
	90			14.17	1.33
	100			13.87	1.47

quency in the EQS 1D-IDCT hardware architecture is lower than the one in the regular 1D-IDCT. Regarding the power dissipation, the EQS 1D-IDCT hardware architecture dissipates between 13.87 mW and 16.84 mW (equivalent to a percentage variation between -9.23% and 10.20%), for BCP values of 100% and 0%, respectively. When compared to the power dissipation of 15.28 mW from the regular 1D-IDCT hardware architecture, the power increase of the EQS 1D-IDCT hardware architecture ranges from -9.25% to 10.2%, for BCP values of 100% and 0%, respectively. The trade-off turning point is the 52% BCP value, where the EQS 1D-IDCT consumes more energy than the regular 1D-IDCT when BCP value is lower than 52%, while it saves energy when BCP values as higher than 52%. Moreover, at the 52% BCP value turning point the BD-Rate increase is 0.621%.

Table 20 shows a comparison between the developed regular 1D-IDCT hardware architecture and related works found in the literature. Table 20 presents the blocks sizes each work can process, the technology used in the synthesis, the equivalent area, the frequency, the dissipated power, and the EPS* value, which divides the Energy Per Sample value by the number of block sizes that can be processed by the works listed in Table 20.

The regular 1D-IDCT occupies an area equivalent to 94,366 gates, which is 2.35 times larger than YAO et al. (2015) and 3.79 times larger than REDDY; RAJABAI; SIVANANTHAM (2017). The developed regular 1D-IDCT hardware architecture presents the maximum operation frequency among the works in Table 19, being at least 1.7 times larger than the biggest maximum operation frequency (JRIDI; MEHER, 2016) of the related works. Moreover, the developed regular 1D-IDCT hardware architecture dissipates slightly more power than JRIDI; MEHER (2016) and more than four times more power than REDDY; RAJABAI; SIVANANTHAM (2017). It is important to point

Table 20 – Comparison of the regular and EQS 1D-IDCT with related work

	Block Sizes	Techno-logy	Area (gates)	Frequency (MHz)	Power (mW)	EPS* (pJ)
Regular 1D-IDCT	4,8,16,32	45nm	94,366	737.46	16.84	0.16
(REDDY; RAJABAI; SIVANANTHAM, 2017)	32	90nm	24,927	370	4.01	0.34
(JRIDI; MEHER, 2016)	4,8,16,32	FPGA	1760	421.94	16.71	0.31
(CONCEICAO et al., 2015)	4	FPGA	152	78.31	N/D	N/A
(YAO et al., 2015)	16,32	65nm	40,100	500	N/D	N/A
(KILANY et al., 2015)	4,8,16,32	65nm	98,500	500	N/D	N/A

out that the developed 1D-IDCT hardware architecture operates at a higher frequency and present higher throughput, which leads to higher toggling rate in the hardware design nets. Thus, the power dissipation is expected to be larger, and REDDY; RAJABAI; SIVANANTHAM (2017) presents a much simpler hardware architecture developed to process only 32-point 1D-IDCT, while the developed regular 1D-IDCT hardware architecture not only processes all 1D-IDCT sizes, it can process constantly 32-samples per cycle, regardless the block size to which each of the 32 input samples belong to. Regarding energy per processed sample normalized by the quantity of block sizes the work can process, the developed 1D-IDCT hardware architecture consumes 0.16 pJ per processed sample, while REDDY; RAJABAI; SIVANANTHAM (2017) consumes 0.34 pJ and JRIDI; MEHER (2017) consumes 0.31 pJ per processed sample.

Table 21 presents the number of CM processed by each work listed in Table 20 and the BD-Rate impact of such throughput limitation on the real-time encoding of HD 1080p videos at 30fps and UHD 4K videos at 60fps.

Table 21 – BD-Rate impact of throughput limitation of works listed in Table 20

	HD 1080p @ 30fps		UHD 4K @ 60fps	
	Proc. Rate	BD-Rate* (%)	Proc. Rate	BD-Rate* (%)
Regular 1D-IDCT	252.90	0	31.62	3.85
(REDDY; RAJABAI; SIVANANTHAM, 2017)	126.89	0	15.87	15.03
(JRIDI; MEHER, 2016)	144.70	0	18.09	13.07
(CONCEICAO et al., 2015)	13.43	19.35	1.68	57.93
(YAO et al., 2015)	140.60	0	17.57	13.45
(KILANY et al., 2015)	171.47	0	21.43	9.45

* estimated based on section 3.2

According to Table 21, the developed hardware architectures have the highest number of CM among the related works, and, therefore, the smallest BD-Rate impact due to the throughput limitation. Almost all of the related works surpasses the 101 CM processing requirement for real-time encoding of HD 1080p videos at 30fps, with the

exception of the 13.43 CM capability for CONCEICAO et al. (2015) which leads to a BD-Rate increase of 19.35%. Regarding UHD4K@60fps videos, all of the works listed in Table 17 are estimated to have BD-Rate increase due to throughput limitation. The developed regular 1D-IDCT hardware architecture has the lowest BD-Rate increase of 3.85%, which is at least half of the smallest BD-Rate increase of the related works – 9.45% for KILANY et al. (2015).

5.3 Direct and Inverse Quantization hardware syntheses results

Subsection 4.3 presents a project space exploration on the direct and inverse quantization in the search for the optimal trade-off between coding efficiency, power efficiency, and throughput. Table 22 shows the comparison of the developed architectures for the direct quantization using flat-quantization (Flat) or frequency-dependent quantization (FDQ) and using multiplierless constant multiplication (MCM) or generic multiplication (GM). Table 22 shows the results, in number of gates, the maximum operation frequency, in MHz, the power dissipated, in mW, for five targets: maximum, HD 1080p videos at 30fps, UHD 4K videos at 30fps, UHD 4K videos at 60fps, and UHD 8K videos at 60fps.

Table 22 – Synthesis results of the direct quantization hardware design space exploration

Flat/FDQ Quantization	Flat	Flat	FDQ	FDQ
MCM/Generic multiplication	Generic	MCM	Generic	MCM
Area (gates)	285,256	423,721	290,809	3,353,567
Maximum Frequency (MHz)	816.99	741.84	888.10	727.80
Power (mW) – Max. Freq.	163.80	191.88	152.13	1,896.89
Power (mW) – HD 1080p@30fps	1.15	1.01	1.10	10.63
Power (mW) – UHD 4K@60fps	3.19	3.33	3.43	25.36

The FDQ implementation using MCM occupies an area equivalent to 3,353,567 gates and is 7.91 times larger than the flat quantization implementation using MCM and over 11 times larger than any of the direct quantization implementations using GM. The FDQ implementation using GM can operate at the highest frequency – 888.10 MHz – among all the implementations, which is 22% higher than the FDQ implementation using MCM, the lowest frequency.

The implementation dissipating less power depends on the target. The FDQ implementation with GM is the most power efficient in its maximum frequency, dissipating 152.14 mW of power, 12 times less power than the FDQ MCM implementation. When targeting HD 1080p videos at 30fps, the flat-quantization using MCM dissipates the lowest power, only 9.5% of the power dissipated by FDQ MCM implementation. The flat-quantization with GM dissipates the least amount of power when targeting UHD 4K

videos at 60fps, 88% lower than the power dissipation on FDQ MCM implementation.

Table 23 shows a comparison of the direct flat quantization using MCM hardware architecture with the DIAS; ROMA; SOUSA (2015) one. The flat quantization using MCM hardware architecture is chosen because it presents the smallest power dissipation for the same target as DIAS; ROMA; SOUSA (2015), the target for real-time encoding of UHD 4K videos at 30fps. There is only one comparison due to the lack of HEVC quantization architectures in the literature, to the best of my knowledge.

Table 23 – Comparison of the flat/MCM DQ-SSC with related work (DIAS; ROMA; SOUSA, 2015)

	Techno-logy	Area (gates)	Frequency (MHz)	Power (mW)	EPS (pJ)
Flat/MCM DQ-SSC	45nm	13,241	741.84	1.93	5.17
(DIAS; ROMA; SOUSA, 2015)	90nm	3,254	694.4	3.9	10.43

The area of the Flat/MCM DQ-SSC is four times larger than the area in DIAS; ROMA; SOUSA (2015). Nonetheless, the flat/MCM DQ-SSC hardware architecture can operate at a frequency 1.07 times larger, dissipates 2.02 times less power, and is 2.02 times more energy efficient than the related work (DIAS; ROMA; SOUSA, 2015).

Table 24 shows a comparison of the Direct Quantization Single Sample Core (DQ-SSC) using flat quantization and MCM hardware architecture with the related work (DIAS; ROMA; SOUSA, 2015). The single sample core (SSC) implementation is used in the comparison because it processes only one sample at the time like the related work (DIAS; ROMA; SOUSA, 2015). The flat quantization using MCM implementation is chosen because it presents the smallest power dissipation for the same target as (DIAS; ROMA; SOUSA, 2015), the target for real-time encode of UHD 4K videos at 30fps. There is only one comparison due to the lack of more HEVC quantization architectures in the literature, to the best of our knowledge.

Table 24 shows the comparison of the developed architectures for the direct quantization using Flat or FDQ quantization and using MCM or GM. Table 24 shows the results of area, maximum operation frequency, and the power dissipated, for the same previous targets.

As it could be seen on the direct quantization synthesis results, the FDQ/MCM implementation occupies an area 2.22 times larger than the flat/GM implementation, which occupies the smallest area. The implementation with the highest maximum frequency is the FDQ/GM implementation, which is 22.52% higher than the flat/MCM implementation (the lowest frequency). Regarding power dissipation, the FDQ/GM implementation has the lowest power dissipation in maximum frequency, while the flat/GM dissipates less power than the other implementations for the target frequencies for real-time encoding of HD 1080p at 30fps and UHD 4K at 60 fps.

Table 24 – Synthesis results of the inverse quantization hardware design space exploration

Flat/FDQ Quantization	Flat	Flat	FDQ	FDQ
MCM/Generic multiplication	Generic	MCM	Generic	MCM
Area (gates)	166,884	206,959	170,684	370,758
Maximum Frequency (MHz)	1,333.33	1,113.59	1,358.70	1,194.74
Power (mW) – Max. Freq.	33.70	95.16	31.15	110.19
Power (mW) – HD 1080p@30fps	0.69	0.90	0.87	1.10
Power (mW) – UHD 4K@60fps	1.40	1.86	1.42	2.56

Table 25 shows a comparison of the Inverse Quantization Single Sample Core (IQ-SSC) using flat quantization and generic multiplication (GM) hardware architecture with the DIAS; ROMA; SOUSA (2015) one. The flat quantization using GM hardware architecture is chosen because it presents the smallest power dissipation for the same target as (DIAS; ROMA; SOUSA, 2015), the target for real-time encode of UHD 4K videos at 30fps.

Table 25 – Comparison of the flat/GM IQ-SSC with related work (DIAS; ROMA; SOUSA, 2015)

	Technology	Area (gates)	Frequency (MHz)	Power (mW)	EPS (pJ)
Flat/GM implementation	45nm	166,884	1,333.33	1.00	2.68
(DIAS; ROMA; SOUSA, 2015)	90nm	2,662	1,041.7	3.9	10.43

The IQ-SSC occupies the double of the area of the related work (DIAS; ROMA; SOUSA, 2015), while the IQ-SSC presents a maximum frequency 28% higher, a power dissipation 3.9 times smaller, and an energy efficiency 3.9 larger than the related work (DIAS; ROMA; SOUSA, 2015).

5.4 Integrated Direct/Inverse Quantization synthesis results

This subsection presents the synthesis results of the Integrated Direct/Inverse Quantization (IDIQ) and the Parallel Integrated Direct/Inverse Quantization (PIDIQ) hardware architectures and compares such results with related works. Table 26 shows the synthesis results of the IDIQ and PIDIQ hardware architecture, detailing the equivalent area, the maximum operating frequency, and the power dissipation, using the same previous targets.

The PIDIQ hardware architecture occupies an area equivalent to 341,380 gates, 33.4 times larger than the area occupied by the IDIQ hardware architecture. Both IDIQ and PIDIQ hardware architectures can operate up to 1.68 GHz and dissipate 12.21 mW and 369.37 mW of power under the maximum frequency, respectively. The dissipation

Table 26 – IDIQ and PIDIQ syntheses results

Parameter	IDIQ	PIDIQ
Area (gates)	10,221	341,380
Maximum Frequency (MHz)	1,683.94	1,679.51
Power (mW) – Max. Freq. (1.68 GHz)	12.21	369.37
Power (mW) – HD 1080p@30fps (2.92 MHz)	0.42	4.57
Power (mW) – UHD 4K@60fps (23.33 MHz)	1.84	19.96

power estimation was also performed targeting real-time encoding of HD 1080p videos at 30fps and UHD 4K at videos 60 fps. The IDIQ hardware architecture dissipates 0.42 mW and the PIDIQ hardware architecture dissipates 4.57 mW when targeting real-time encoding of HD 1080p videos at 30fps. For the target of real-time encoding of UHD 4K videos at 60fps, the IDIQ and PIDIQ hardware architectures dissipate 1.83 mW and 19.66 mW, respectively.

Table 27 shows the comparison of the IDIQ hardware architecture and a concatenation of the direct and inverse quantization modules presented by (DIAS; ROMA; SOUSA, 2015). Table 27 presents the technological node, the equivalent area, the maximum operating frequency, the dissipated power when processing UHD 4K videos at 30fps in real time, and the EPS rate.

Table 27 – Comparison of the IDIQ hardware architectures with related work (DIAS; ROMA; SOUSA, 2015)

		(DIAS; ROMA; SOUSA, 2015)		
Parameter	IDIQ	Direct	Inverse	Direct/Inverse
Technology	45nm	90nm		
Area (gates)	10,221	3,254	2,662	5,916
Frequency (MHz)	1,683.94	1,041.7	694.4	694.4
Power (mW)	1.12	3.9	3.9	7.8
EPS (pJ)	2.99	10.43	10.43	20.86

The area of the IDIQ hardware architecture is 72% larger, the maximum frequency is 2.43 times higher, the power dissipation is 7 times lower than the combined area of the direct quantization and inverse quantization modules of the related work (DIAS; ROMA; SOUSA, 2015). Also, the IDIQ hardware architecture is 7 times more energy efficient than the combined direct- and inverse-quantization modules of the related work (DIAS; ROMA; SOUSA, 2015).

5.5 Final Considerations

The developed architectures usually occupy larger areas than the respective related works. Nonetheless, they can operate at higher frequencies and consequently

can achieve much higher throughput than the related works. Moreover, the developed architectures are more energy efficient than the related works.

6 CONCLUSION

This master thesis has presented multiple energy/quality-aware hardware designs for the residual coding loop components of the HEVC standard. In addition to the usual low-power focus of the hardware designs in related works, the hardware designs presented in this work focus on meeting the real throughput requirements of the residual coding loop, a design constraint overlooked by the vast majority of the related works.

The constraints of the hardware solutions for the HEVC RCL components were based on the experiments presented in this work. An analysis of the throughput requirements, performed using all CTC videos and configuration profiles, indicates that the HEVC reference software tests on average 101 CM for the DCT and 130 CM for the quantization to maximize the coding efficiency. Further analysis shows an impact of hardware designs throughput limitations on the coded video size of up to 55.70% and 125.87% for single-CM DCT and quantization hardware architectures, respectively. The BD-Rate impact of the Frequency-Dependent Quantization coding tool, available on the HEVC, was also analyzed, showing a range from 7.94% BD-Rate reduction to 0.64% BD-Rate increase, with an average of 0.92% BD-Rate reduction.

The hardware solution developed in this work are an FFT-inspired HEVC 1D-DCT hardware architecture, an energy/quality scalable HEVC 1D-IDCT hardware architecture, a set of direct- and inverse-quantization hardware architectures, and an integrated direct/inverse quantization hardware architecture.

The FFT-inspired HEVC 1D-DCT achieves low power consumption (12.33 mW) at a high operational frequency (2.54 GHz), due to the high hardware reuse. The HEVC 1D-IDCT was implemented in two versions – the regular HEVC 1D-IDCT and the energy/quality scalable (EQS) HEVC 1D-IDCT – and achieves high frequencies (737.46 MHz and 720.46 MHz, respectively). The regular HEVC 1D-IDCT dissipates low power (15.28 mW) and the EQS HEVC 1D-IDCT can reduce it up to 10% with an impact of up to 1.47% BD-Rate increase. The set of direct- and inverse-quantization hardware architectures explore the project space through the combination of the comparisons of the flat- or frequency-dependent-quantization (FDQ) and the multiplierless constant multiplication (MCM) or generic multiplication (GM). The optimal setup for the direct

quantization is the FDQ version using MCM, with a power dissipation of 152.13 mW at 888.10 MHz, its maximum operating frequency, and the optimal setup for the inverse quantization is also the FDQ version using MCM, with a power dissipation of 31.15 mW at 1.36 GHz. The FDQ/MCM implementation of the direct- and inverse-quantization presented the highest operation frequency and the lowest power dissipation. The parallel integrated direct/inverse (PIDIQ) quantization reaches high frequency, up to 1.68 GHz.

The hardware architectures developed in this work achieve the highest frequencies and the highest energy efficiency among their related works. The FFT-inspired HEVC 1D-DCT hardware architecture achieves frequency 3.55 times higher, and an energy efficiency at least 80% larger than its related works. The regular 1D-IDCT achieves 70% higher operational frequency and is twice more energy efficient than the related works. The direct quantization can operate at a frequency 6.8% higher and is twice as energy efficient as its related work. The inverse quantization operates to a frequency 28% higher and is four times more energy efficient than its related works. The integrated direct/inverse quantization (IDIQ) hardware architecture achieves a frequency 2.42 times higher than its related work and is seven times more energy efficient than its related work.

Moreover, the throughput achieved by the developed HEVC RCL components hardware architectures allows them to have zero BD-Rate impact due to throughput limitation, while all related works present BD-Rate impacts between 4.23% and 134.71%.

Therefore, the architectures in this work achieve the proposed objectives, as they do not impact on BD-Rate due to throughput limitations while being more energy efficient than all of their respective related works.

6.1 Future works

The developed HEVC RCL components hardware architectures showed high throughput and low power features. Nonetheless, there is still room for improvement.

The 1D-IDCT hardware architectures – regular and EQS – can have their throughput increased and power dissipation reduced by implementing it based on the IFFT algorithm. Moreover, the EQS 1D-IDCT hardware architecture can use a coarser resolution on its Bypassed Columns Percentage (BCP) value, from the current 0.1% step to 1% or 5% step, which will simplify the bypass control blocks, reducing the dissipating power overhead due to the bypass control blocks.

The PIDIQ hardware architecture uses collections of LUTs to simplify the modulus operation. Since the QP range value used is a subset of the 0 to 51 specified range, the power dissipation of the PIDIQ hardware architecture can be reduced and its maximum frequency can be increased.

Moreover, the 1D-DCT and 1D-IDCT hardware architectures can be used to implement 2D-DCT and 2D-IDCT hardware architectures, respectively. Furthermore, a precise and approximate complete RCL hardware architectures can be implemented with the developed RCL components hardware architectures.

REFERENCES

AGOSTINI, L. **Desenvolvimento de arquiteturas de alto desempenho dedicadas a compressão de vídeo segundo o padrão H.264/AVC**. 2007. 172p. Tese — PPGC/UFRGS, Porto Alegre.

ANSARI, A.; MANSOURI, A.; AHAITOUF, A. An efficient VLSI architecture for integer DCT in HEVC standard. **Proc. IEEE/ACS International Conference of Computer Systems and Applications**, Agadir, Morocco, 2016.

APPLE. **iPhone 7 - Technical Specifications - Apple**. Disponível em: <https://www.apple.com/iphone-7/specs/>. Acesso em: 17 jan. 2018.

BARRET, P. Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor. **Proc. Advances in Cryptology — CRYPTO' 86**, Berlin, 1987.

BASIRI, M. A. M.; MAHAMMAD, N. High performance integer DCT architectures for HEVC. **Proc. International Conference on VLSI Design**, Hyderabad, India, jan 2017.

BJONTEGAARD, G. Calculation of average PSNR difference between RD-curves. , Documento VCEG-M33, abr 2001.

BONATTO, L. V. M. et al. Low-power multi-size HEVC DCT architecture proposal for QFHD video processing. **Proc. IEEE Symposium on Integrated Circuits and Systems Design**, Fortaleza, Brazil, 2017.

BOSSEN, F. Common test condition and software reference configuration. , 12ª Reunião do JCT-VC, jan 2013.

BRAATZ, L.; AGOSTINI, L.; ZATT, B.; PORTO, M. A Multiplierless Parallel HEVC Quantization Hardware for RealTime UHD 8K Video Coding. **Proc. IEEE Intenational Symposium on Circuits And Systems**, Baltimore, mai 2017.

BUDAGAVI, M.; FULDSETH, A.; BJONTEGAARD, G. HEVC Transform and Quantization. In: SZE, V.; BUDAGAVI, M.; SULLIVAN, G. J. (Ed.). **High Efficiency Video Coding (HEVC)**: Algorithms and architectures. New York: Springer, 2014. p.141–170.

BUDAGAVI, M.; FULDSETH, A.; BJONTEGAARD, G.; SZE, V. Core transform design in the High Efficiency Video Coding (HEVC) standard. **IEEE Journal of Selected Topics in Signal Processing**, Newark, v.7, dez 2013.

CADENCE. **Encounter RTL Compiler**. Disponível em: <https://www.cadence.com/content/cadence-www/global/en_US/training/all-courses/84441.html>. Acesso em: 15 jan. 2018.

CHATTERJEE, S.; SARAWADEKAR, K. P. A low cost, constant throughput and reusable 8x8 DCT architecture for HEVC. **Proc. International Midwest Symposium on Circuits and Systems**, Abu Dhabi, UAE, mar 2017.

CISCO. Cisco visual networking index: Forecast and methodology, 2016-2021. , [S.l.], jun 2017.

COELHO, D. F. G.; CINTRA, R. J.; DIMITROV, V. S. Efficient Computation of the 8-point DCT via Summation by Parts. **Journal of Signal Processing Systems**, [S.l.], Aug 2017.

CONCEICAO, R. et al. Hardware Design of Fast HEVC 2-D IDCT Targeting Real-Time UHD 4K Applications. **Proc. IEEE Latin America Symposium on Circuits Systems Design**, Montevideo, Uruguay, 2015.

COOLEY, J. W.; TUKEY, J. W. An algorithm for the machine calculation of complex Fourier series. **Mathematics of Computation**, Tamilnadu, India, v.19, p.297–301, abr 1965.

CORREA, G.; ASSUNCAO, P.; AGOSTINI, L.; CRUZ, L. A. S. Performance and Computational Complexity Assessment of High-Efficiency Video Encoders. **IEEE Transactions on Circuits and Systems for Video Technology**, Newark, v.22, oct 2012.

DAVORE, J. L. **Probability and Statistics for Engineering and the Sciences**. Boston, MA: Cengage Learning, 2011. 407-408p.

DIAS, T.; ROMA, N.; SOUSA, L. High performance IP core for HEVC Quantization. **Proc. IEEE Intenational Symposium on Circuits And Systems**, Lisboa, mai 2015.

FLYNN, D.; BOSSEN, F.; SHARMAN, K.; SUHRING, K. HEVC Reference Software Manual. , Munique, jul 2011.

GHANBARI, M. **Standard Codecs**: Image Compression to Advanced Video Coding. Stevenage: The Institution of Electrical Engineers, 2003.

GOEBEL, J. et al. An HEVC multi-size DCT hardware with constant throughput and supporting heterogeneous CUs. **Proc. IEEE Intenational Symposium on Circuits And Systems**, Montreal, mai 2016.

HAQUE, M.; TABATABAI, A.; MORIGAMI, Y. HVS model based default quantization matrices. **Joint Collaborative Team on Video Coding, Document JCT-VC/G880**, Geneva, Switzerland, nov 2011.

HUNG, C.-Y.; LANDMAN, P. Compact inverse discrete cosine transform circuit for MPEG video decoding. **Proc. IEEE International Workshop on Signal Processing Systems**, Leicester, 1997.

ITU-T. Advanced video coding for generic audiovisual services. **Recommendation H.264**, Geneva, Switzerland, mai 2003.

ITU-T. High efficiency video coding. **Series H: Audiovisual and multimedia systems**, [S.l.], abr 2015.

JESKE, R. **Otimizações Algorítmicas e Desenvolvimento Arquitetural para as DCTs do HEVC**. 2013. 110p. Dissertação de Mestrado — PPGC/UFPEL, Pelotas.

JONES, D. L. Decimation-in-time (DIT) radix-2 FFT. **OpenStax-CNX module: ml2016**, Montreal, mai 2016.

JRIDI, M.; MEHER, P. K. Scalable approximate DCT architecture for efficient HEVC-compliant video coding. **IEEE Transactions on Circuits and Systems for Video Technology**, Newark, v.27, jul 2016.

KILANY, A.; ABDELRASOUL, M.; SHALABY, A.; SAYED, M. S. A reconfigurable 2-D IDCT architecture for HEVC encoder/decoder. **Proc. International Conference on Microelectronis**, Casablanca, Morocco, 2015.

LEINEMA, J.; HAN, W.-J. Intra-picutre prediction on HEVC. In: SZE, V.; BUDAGAVI, M.; SULLIVAN, G. J. (Ed.). **High Efficiency Video Coding (HEVC)**: Algorithms and architectures. New York: Springer, 2014. p.91–112.

MARKOV, I. L. Limits on fundamental limits to computation. **Nature**, Londres, v.512, ago 2014.

MASERA, M.; MARTINA, M.; MASERA, G. Adaptive approximate DCT architectures for HEVC. **IEEE Transactions on Circuits and Systems for Video Technology**, Newark, v.27, jul 2017.

MODELSIM. **ModelSim(R) HDL simulator**. Disponível em: <https://www.mentor.com/company/higher_ed/modelsim-student-edition>. Acesso em: 10 jan. 2018.

MONTGOMERY, P. L. Modular multiplication without trial division. **Mathematics of computation**, Rhode Island, v.44, 1985.

NANGATE. **Nangate 45-nm Open Cell Library**. Disponível em: <http://www.nangate.com/?page_id=2325>. Acesso em: 20 jan. 2018.

PLONKA, G.; TASCHE, M. Fast and numerically stable algorithms for discrete cosine transforms. **Linear algebra and its applications**, Newark, v.394, jan 2005.

PRAGNELL, L.; SANCHEZ, V. Adaptive quantization matrices for HD and UHD display resolutions in scalable HEVC. **Proc. Data Compression Conference**, Snowbird, 2016.

QUALCOMM. **Snapdragon 835 Mobile Platform**. Disponível em: <https://www.qualcomm.com/products/snapdragon/processors/835>. Acesso em: 17 jan. 2018.

REDDY, V. A.; RAJABAI, C. P.; SIVANANTHAM, S. Hardware Implementation for the 32x32 IDCT of High-Efficiency Video Coding. **Asian Journal of Applied Science and Technology**, Tamilnadu, India, v.1, p.127–130, jun 2017.

RENDA, G.; MASERA, M.; MARTINA, M.; MASERA, G. Approximate Arai DCT architecture for HEVC. **Proc. 2017 New Generation of CAS**, Genova, Italy, set 2017.

SAMSUNG. **Exynos 9 Series - S5E9810 | Samsung Semiconductor Global Website**. Disponível em: <http://www.samsung.com/semiconductor/processor/mobile-processor/S5E9810/>. Acesso em: 2018-01-17.

SAXENA, A.; FERNANDES, F. C. DCT/DST-based transform coding for intra prediction in image/video coding. **IEEE Transactions on Image Processing**, Newark, v.22, 2013.

SCHWARZ, H.; SCHIERL, T.; MARPE, D. Block structures and parallelism features in HEVC. In: SZE, V.; BUDAGAVI, M.; SULLIVAN, G. J. (Ed.). **High Efficiency Video Coding (HEVC)**: Algorithms and architectures. New York: Springer, 2014. p.49–90.

SELESNICK, I.; SCHULLER, G. The Discrete Fourier Transform. In: (EDITOR), K. R. R.; (EDITOR), P. C. Y. (Ed.). **The Transform and Data Compression Handbook**. Boca Raton: CRC Press, 2000. p.37–78.

SHIN, Y.; SEOMUN, J.; CHOI, K.-M.; SAKURAI, T. Power gating: Circuits, design methodologies, and best practice for standard-cell VLSI designs. **ACM Transactions on Design Automation of Electronic Systems**, New York, v.15, set 2010.

SMITH, S. W. **The Scientists and Engineers Guide to Digital Signal Processing**. San Diego: California Technical Publishing, 1997.

SULLIVAN, G.; OHM, J.-R.; WANG, W.-J.; WIEGAND, T. Overview of the High Efficiency Video Coding (HEVC) Standard. **IEEE Transactions on Circuits and Systems for Video Technology**, Newark, v.12, dez 2012.

YAO, Z. et al. Area and throughput efficient IDCT/IDST architectures for HEVC standard. **Proc. IEEE International Symposium on Circuits and Systems**, Melbourne, Australia, 2015.

ZHOU, C.; ZHOU, F.; CHEN, Y. Spatio-temporal correlation-based fast coding unit depth decision for high efficiency video coding. **Journal of Electronic Imaging**, Bellingham, v.22, 2013.

APPENDIX A TRANSFORM MATRICES

This appendix shows the 4×4-, 8×8-, and 16×16-DCT integer approximation matrices in Tables 28 to 30. Table 31 shows the left half of the 32×32 DCT integer approximation matrix, due to its size and the symmetric properties of DCT matrices. And Table 32 presents the 4×4 DST integer approximation matrix.

Table 28 – 4×4 DCT integer approximation

64	64	64	64
83	36	-36	-83
64	-64	-64	64
36	-83	83	-36

Table 29 – 8×8 DCT integer approximation

64	**64**	**64**	**64**	64	64	64	64
89	75	50	18	-18	-50	-75	-89
83	**36**	**-36**	**-83**	-83	-36	36	83
75	-18	-89	-50	50	89	18	-75
64	**-64**	**-64**	**64**	64	-64	-64	64
50	-89	18	75	-75	-18	89	-50
36	**-83**	**83**	**-36**	-36	83	-83	36
18	-50	75	-89	89	-75	50	-18

Table 30 – 16×16 DCT integer approximation

64	**64**	**64**	**64**	**64**	**64**	**64**	**64**	64	64	64	64	64	64	64	64
90	87	80	70	57	43	25	9	-9	-25	-43	-57	-70	-80	-87	-90
89	**75**	**50**	**18**	**-18**	**-50**	**-75**	**-89**	-89	-75	-50	-18	18	50	75	89
87	57	9	-43	-80	-90	-70	-25	25	70	90	80	43	-9	-57	-87
83	**36**	**-36**	**-83**	**-83**	**-36**	**36**	**83**	83	36	-36	-83	-83	-36	36	83
80	9	-70	-87	-25	57	90	43	-43	-90	-57	25	87	70	-9	-80
75	**-18**	**-89**	**-50**	**50**	**89**	**18**	**-75**	-75	18	89	50	-50	-89	-18	75
70	-43	-87	9	90	25	-80	-57	57	80	-25	-90	-9	87	43	-70
64	**-64**	**-64**	**64**	**64**	**-64**	**-64**	**64**	64	-64	-64	64	64	-64	-64	64
57	-80	-25	90	-9	-87	43	70	-70	-43	87	9	-90	25	80	-57
50	**-89**	**18**	**75**	**-75**	**-18**	**89**	**50**	50	89	-18	-75	75	18	-89	50
43	-90	57	25	-87	70	9	-80	80	-9	-70	87	-25	-57	90	-43
36	**-83**	**83**	**-36**	**-36**	**83**	**-83**	**36**	36	-83	83	-36	-36	83	-83	36
25	-70	90	-80	43	9	-57	87	-87	57	-9	-43	80	-90	70	-25
18	**-50**	**75**	**-89**	**89**	**-75**	**50**	**-18**	-18	50	-75	89	-89	75	-50	18
9	-25	43	-57	70	-80	87	-90	90	-87	80	-70	57	-43	25	-9

Table 31 – 32×32 DCT integer approximation

64	**64**	**64**	**64**	**64**	**64**	**64**	**64**	**64**	**64**	**64**	**64**	**64**	**64**	**64**	**64**
90	90	88	85	82	78	73	67	61	54	46	38	31	22	13	4
90	**87**	**80**	**70**	**57**	**43**	**25**	**9**	**-9**	**-25**	**-43**	**-57**	**-70**	**-80**	**-87**	**-90**
90	82	67	46	22	-4	-31	-54	-73	-85	-90	-88	-78	-61	-38	-13
89	**75**	**50**	**18**	**-18**	**-50**	**-75**	**-89**	**-89**	**-75**	**-50**	**-18**	**18**	**50**	**75**	**89**
88	67	31	-13	-54	-82	-90	-78	-46	-4	38	73	90	85	61	22
87	**57**	**9**	**-43**	**-80**	**-90**	**-70**	**-25**	**25**	**70**	**90**	**80**	**43**	**-9**	**-57**	**-87**
85	46	-13	-67	-90	-73	-22	38	82	88	54	-4	-61	-90	-78	-31
83	**36**	**-36**	**-83**	**-83**	**-36**	**36**	**83**	**83**	**36**	**-36**	**-83**	**-83**	**-36**	**36**	**83**
82	22	-54	-90	-61	13	78	85	31	-46	-90	-67	4	73	88	38
80	**9**	**-70**	**-87**	**-25**	**57**	**90**	**43**	**-43**	**-90**	**-57**	**25**	**87**	**70**	**-9**	**-80**
78	-4	-82	-73	13	85	67	-22	-88	-61	31	90	54	-38	-90	-46
75	**-18**	**-89**	**-50**	**50**	**89**	**18**	**-75**	**-75**	**18**	**89**	**50**	**-50**	**-89**	**-18**	**75**
73	-31	-90	-22	78	67	-38	-90	-13	82	61	-46	-88	-4	85	54
70	**-43**	**-87**	**9**	**90**	**25**	**-80**	**-57**	**57**	**80**	**-25**	**-90**	**-9**	**87**	**43**	**-70**
67	-54	-78	38	85	-22	-90	4	90	13	-88	-31	82	46	-73	-61
64	**-64**	**-64**	**64**	**64**	**-64**	**-64**	**64**	**64**	**-64**	**-64**	**64**	**64**	**-64**	**-64**	**64**
61	-73	-46	82	31	-88	-13	90	-4	-90	22	85	-38	-78	54	67
57	**-80**	**-25**	**90**	**-9**	**-87**	**43**	**70**	**-70**	**-43**	**87**	**9**	**-90**	**25**	**80**	**-57**
54	-85	-4	88	-46	-61	82	13	-90	38	67	-78	-22	90	-31	-73
50	**-89**	**18**	**75**	**-75**	**-18**	**89**	**-50**	**-50**	**89**	**-18**	**-75**	**75**	**18**	**-89**	**50**
46	-90	38	54	-90	31	61	-88	22	67	-85	13	73	-82	4	78
43	**-90**	**57**	**25**	**-87**	**70**	**9**	**-80**	**80**	**-9**	**-70**	**87**	**-25**	**-57**	**90**	**-43**
38	-88	73	-4	-67	90	-46	-31	85	-78	13	61	-90	54	22	-82
36	**-83**	**83**	**-36**	**-36**	**83**	**-83**	**36**	**36**	**-83**	**83**	**-36**	**-36**	**83**	**-83**	**36**
31	-78	90	-61	4	54	-88	82	-38	-22	73	-90	67	-13	-46	85
25	**-70**	**90**	**-80**	**43**	**9**	**-57**	**87**	**-87**	**57**	**-9**	**-43**	**80**	**-90**	**70**	**-25**
22	-61	85	-90	73	-38	-4	46	-78	90	-82	54	-13	-31	67	-88
18	**-50**	**75**	**-89**	**89**	**-75**	**50**	**-18**	**-18**	**50**	**-75**	**89**	**-89**	**75**	**-50**	**18**
13	-38	61	-78	88	-90	85	-73	54	-31	4	22	-46	67	-82	90
9	**-25**	**43**	**-57**	**70**	**-80**	**87**	**-90**	**90**	**-87**	**80**	**-70**	**57**	**-43**	**25**	**-9**
4	-13	22	-31	38	-46	54	-61	67	-73	78	-82	85	-88	90	-90

Table 32 – 4×4 DST integer approximation

29	55	74	84
74	74	0	74
84	-29	-74	55
55	-84	74	-29

APPENDIX B QUANTIZATION MATRICES

The quantization matrices (QM) used in the direct quantization are presented in Tables 33 to 38. The matrix used in the direct quantization depends on the remainder of the division of QP by six. Tables 39 to 44 show the matrices used in the inverse quantization, also depending on the remainder of the division of QP by six.

Table 33 – Quantization Matrix 8×8 (QP = 0, 6, 12, 18, 24, 30, 36, 42, 48)

26214	26214	26214	26214	24672	23301	19972	17476
26214	26214	26214	26214	24672	22074	19064	16776
26214	26214	24672	23301	20971	19064	16776	14462
26214	26214	23301	19972	17476	15534	13529	11650
24672	24672	20971	17476	13980	11983	10229	8923
23301	22074	19064	15534	11983	9532	7767	6452
19972	19064	16776	13529	10229	7767	5991	4766
17476	16776	14462	11650	8923	6452	4766	3647

Table 34 – Quantization Matrix 8×8 (QP = 1, 7, 13, 19, 25, 31, 37, 43, 49)

23302	23302	23302	23302	21931	20712	17753	15534
23302	23302	23302	23302	21931	19622	16946	14913
23302	23302	21931	20712	18641	16946	14913	12856
23302	23302	20712	17753	15534	13808	12026	10356
21931	21931	18641	15534	12427	10652	9093	7932
20712	19622	16946	13808	10652	8473	6904	5735
17753	16946	14913	12026	9093	6904	5326	4236
15534	14913	12856	10356	7932	5735	4236	3242

Table 35 – Quantization Matrix 8×8 (QP = 2, 8, 14, 20, 26, 32, 38, 44, 50)

20560	20560	20560	20560	19350	18275	15664	13706
20560	20560	20560	20560	19350	17313	14952	13158
20560	20560	19350	18275	16448	14952	13158	11343
20560	20560	18275	15664	13706	12183	10611	9137
19350	19350	16448	13706	10965	9398	8023	6999
18275	17313	14952	12183	9398	7476	6091	5060
15664	14952	13158	10611	8023	6091	4699	3738
13706	13158	11343	9137	6999	5060	3738	2860

Table 36 – Quantization Matrix 8×8 (QP = 3, 9, 15, 21, 27, 33, 39, 45, 51)

18396	18396	18396	18396	17313	16352	14016	12264
18396	18396	18396	18396	17313	15491	13378	11773
18396	18396	17313	16352	14716	13378	11773	10149
18396	18396	16352	14016	12264	10901	9494	8176
17313	17313	14716	12264	9811	8409	7178	6262
16352	15491	13378	10901	8409	6689	5450	4528
14016	13378	11773	9494	7178	5450	4204	3344
12264	11773	10149	8176	6262	4528	3344	2559

Table 37 – Quantization Matrix 8×8 (QP = 4, 10, 16, 22, 28, 34, 40, 46)

16384	16384	16384	16384	15420	14563	12483	10922
16384	16384	16384	16384	15420	13797	11915	10485
16384	16384	15420	14563	13107	11915	10485	9039
16384	16384	14563	12483	10922	9709	8456	7281
15420	15420	13107	10922	8738	7489	6393	5577
14563	13797	11915	9709	7489	5957	4854	4032
12483	11915	10485	8456	6393	4854	3744	2978
10922	10485	9039	7281	5577	4032	2978	2279

Table 38 – Quantization Matrix 8×8 (QP = 5, 11, 17, 23, 29, 35, 41, 47)

14564	14564	14564	14564	13707	12945	11096	9709
14564	14564	14564	14564	13707	12264	10592	9320
14564	14564	13707	12945	11651	10592	9320	8035
14564	14564	12945	11096	9709	8630	7516	6472
13707	13707	11651	9709	7767	6657	5683	4957
12945	12264	10592	8630	6657	5296	4315	3584
11096	10592	9320	7516	5683	4315	3328	2648
9709	9320	8035	6472	4957	3584	2648	2026

Table 39 – Inverse Quantization Matrix 8×8 (QP = 0, 6, 12, 18, 24, 30, 36, 42, 48)

40	40	40	40	43	45	58	60
40	40	40	40	43	48	55	63
40	40	43	45	50	55	63	73
40	40	45	53	60	68	78	90
43	43	50	60	75	88	103	118
45	48	55	68	88	110	135	163
58	55	63	78	103	135	175	220
60	63	73	90	118	163	220	288

Table 40 – Inverse Quantization Matrix 8×8 (QP = 1, 7, 13, 19, 25, 31, 37, 43, 49)

45	45	45	45	48	51	59	68
45	45	45	45	48	53	62	70
45	45	48	51	56	62	70	82
45	45	51	59	68	76	87	101
48	48	56	68	84	98	115	132
51	53	62	76	98	124	152	183
59	62	70	87	115	152	197	248
68	70	82	101	132	183	248	323

Table 41 – Inverse Quantization Matrix 8×8 (QP = 2, 8, 14, 20, 26, 32, 38, 44, 50)

51	51	51	51	54	57	67	77
51	51	51	51	54	61	70	80
51	51	54	57	64	70	80	92
51	51	57	67	77	86	99	115
54	54	64	77	96	112	131	150
57	61	70	86	112	140	172	207
67	70	80	99	131	172	223	281
77	80	92	115	150	207	281	367

Table 42 – Inverse Quantization Matrix 8×8 (QP = 3, 9, 15, 21, 27, 33, 39, 45, 51)

57	57	57	57	61	64	75	86
57	57	57	57	61	68	78	89
57	57	61	64	71	78	89	103
57	57	64	75	86	96	110	128
61	61	71	86	107	125	146	167
64	68	78	96	125	157	192	232
75	78	89	110	146	192	249	314
86	89	103	128	167	232	314	410

Table 43 – Inverse Quantization Matrix 8×8 (QP = 4, 10, 16, 22, 28, 34, 40, 46)

64	64	64	64	68	72	84	96
64	64	64	64	68	76	88	100
64	64	68	72	80	88	100	116
64	64	72	80	96	108	124	144
68	68	80	96	120	140	164	188
72	76	88	108	140	176	216	260
84	88	100	124	164	216	280	352
96	100	116	144	188	260	352	460

Table 44 – Inverse Quantization Matrix 8×8 (QP = 5, 11, 17, 23, 29, 35, 41, 47)

72	72	72	72	76	81	95	108
72	72	72	72	76	86	99	113
72	72	76	81	90	99	113	131
72	72	81	95	108	122	140	162
76	76	90	108	135	158	185	212
81	86	99	122	158	198	243	293
95	99	113	140	185	243	315	396
108	113	131	162	212	293	396	518

APPENDIX C BD-RATE IMPACT EVALUATION OF THE RCL DATA

As mentioned in section 3.2, the data in Table 9 is an average of three videos in each resolution. This appendix details the reduction percentage on the transform and quantization processing rates and BD-Rate variation for each video composing the experiments in section 3.2. Tables 45 and 46 present the same structure as Table 9. The TB quadtree size range column presents the TB range of allowed TB sizes during the simulations, QP column shows the QP value used on the simulation, Tr(%) and Qt(%) columns show the reduction percentage on the transform processing rate and on the quantization processing rate, respectively, and the BR(%) column represent the BD-Rate variation caused by the reduction in the TB quadtree depth.

Table 45 – Impact of reducing the TB size range on BD-Rate, and number of processed coding modes of the DCT and the quantization in HD 1080p videos

TB quadtree size range	QP	BQTerrace			Cactus			Kimono		
		Tr (%)	Qt (%)	BR (%)	Tr (%)	Qt (%)	BR (%)	Tr (%)	Qt (%)	BR (%)
4 x 4 to 4 x 4	22	52.52	34.50	8.89	55.20	37.43	21.97	55.52	37.87	42.24
	27	57.02	36.90	8.89	58.78	40.01	21.97	57.02	40.61	42.24
	32	58.03	36.88	8.89	60.35	40.87	21.97	58.03	43.68	42.24
	37	58.60	37.37	8.89	62.07	42.43	21.97	58.60	46.35	42.24
4 x 4 to 8 x 8	22	37.18	27.47	3.71	38.83	29.07	17.83	37.82	28.37	17.83
	27	41.90	31.15	3.71	41.38	31.04	17.83	40.04	30.24	17.83
	32	41.27	29.98	3.71	42.20	31.45	17.83	41.78	31.54	17.83
	37	40.16	28.67	3.71	42.95	31.99	17.83	43.21	32.60	17.83
4 x 4 to 16 x 16	22	18.89	15.09	1.17	19.13	15.35	4.64	18.47	14.78	4.64
	27	22.96	18.50	1.17	21.00	16.84	4.64	20.11	16.25	4.64
	32	22.06	17.50	1.17	21.68	17.33	4.64	21.64	17.57	4.64
	37	20.85	16.28	1.17	21.84	17.40	4.64	22.27	18.03	4.64

Table 46 – Impact of reducing the TB size range on BD-Rate, and number of processed coding modes of the DCT and the quantization in UHD 4K videos

TB quadtree size range	QP	Beauty			Jockey			Shake N' Dry		
		Tr (%)	Qt (%)	BR (%)	Tr (%)	Qt (%)	BR (%)	Tr (%)	Qt (%)	BR (%)
4 x 4 to 4 x 4	22	48.57	31.21	0.26	48.30	31.13	7.77	48.15	30.79	3.82
	27	48.60	31.26	0.26	48.55	31.40	7.77	48.33	30.97	3.82
	32	48.73	31.41	0.26	48.92	31.79	7.77	48.47	31.12	3.82
	37	49.06	31.74	0.26	49.05	31.87	7.77	48.66	31.38	3.82
4 x 4 to 8 x 8	22	33.76	24.48	0.10	33.51	24.34	3.84	33.43	24.23	1.19
	27	33.82	24.54	0.10	33.82	24.65	3.84	33.61	24.40	1.19
	32	33.87	24.61	0.10	34.10	24.94	3.84	33.69	24.48	1.19
	37	34.17	24.91	0.10	34.21	25.04	3.84	33.77	24.57	1.19
4 x 4 to 16 x 16	22	16.66	13.11	0.12	16.59	13.07	2.41	16.59	13.08	0.84
	27	16.66	13.12	0.12	16.75	13.22	2.41	16.65	13.13	0.84
	32	16.71	13.18	0.12	16.91	13.39	2.41	16.72	13.20	0.84
	37	16.86	13.33	0.12	16.98	13.47	2.41	16.75	13.23	0.84

APPENDIX D QSTEP CONSTANT VALUES

Table 47 shows the all the QStep possible values, according to all valid QP values and TB size.

Table 47 – QStep constant values

QP value	TB size 4	8	16	32	QP value	TB size 4	8	16	32
0	20	10	5	3	26	408	204	102	51
1	23	11	6	3	27	456	228	114	57
2	26	13	6	3	28	512	256	128	64
3	29	14	7	4	29	576	288	144	72
4	32	16	8	4	30	640	320	160	80
5	36	18	9	5	31	720	360	180	90
6	40	20	10	5	32	816	408	204	102
7	45	23	11	6	33	912	456	228	114
8	51	26	13	6	34	1024	512	256	128
9	57	29	14	7	35	1152	576	288	144
10	64	32	16	8	36	1280	640	320	160
11	72	36	18	9	37	1440	720	360	180
12	80	40	20	10	38	1632	816	408	204
13	90	45	23	11	39	1824	912	456	228
14	102	51	26	13	40	2048	1024	512	256
15	114	57	29	14	41	2304	1152	576	288
16	128	64	32	16	42	2560	1280	640	320
17	144	72	36	18	43	2880	1440	720	360
18	160	80	40	20	44	3264	1632	816	408
19	180	90	45	23	45	3648	1824	912	456
20	204	102	51	26	46	4096	2048	1024	512
21	228	114	57	29	47	4608	2304	1152	576
22	256	128	64	32	48	5120	2560	1280	640
23	288	144	72	36	49	5760	2880	1440	720
24	320	160	80	40	50	6528	3264	1632	816
25	360	180	90	45	51	7296	3648	1824	912

APPENDIX E PUBLICATIONS

During the development of this master thesis, two papers were accepted for publication on relevant conferences:

- **Title:** A Multiplierless Parallel HEVC Quantization Hardware for Real Time UHD 8K Video Coding
  Event: 2017 IEEE International Symposium on Circuits and Systems (ISCAS)
  Authors: Luciano Braatz, Luciano Agostini, Bruno Zatt, and Marcelo Porto
  Qualis: A1

- **Title:** High-Throughput and Low-Power Integrated Direct/Inverse HEVC Quantization Hardware Design
  Event: 2018 IEEE International Symposium on Circuits and Systems (ISCAS)
  Authors: Luciano Braatz, Daniel Palomino, Bruno Zatt, Luciano Agostini, and Marcelo Porto
  Qualis: A1