# A COST SHARED QUANTIZATION ALGORITHM AND ITS IMPLEMENTATION FOR MULTI - STANDARD VIDEO CODECS

A Thesis Submitted to the College of

Graduate Studies and Research

In Partial Fulfillment of the Requirements

For the Degree of Master of Science

In the Department of Electrical and Computer Engineering

University of Saskatchewan

Saskatoon, Saskatchewan, Canada

By

MOUSUMI DAS

# ABSTRACT

The current trend of digital convergence creates the need for the video encoder and decoder system, known as codec in short, that should support multiple video standards on a single platform. In a modern video codec, quantization is a key unit used for video compression. In this thesis, a generalized quantization algorithm and hardware implementation is presented to compute quantized coefficient for six different video codecs including the new developing codec High Efficiency Video Coding (HEVC). HEVC, successor to H.264/MPEG-4 AVC, aims to substantially improve coding efficiency compared to AVC High Profile. The thesis presents a high performance circuit shared architecture that can perform the quantization operation for HEVC, H.264/AVC, AVS, VC-1, MPEG- 2/4 and Motion JPEG (MJPEG). Since HEVC is still in drafting stage, the architecture was designed in such a way that any final changes can be accommodated into the design. The proposed quantizer architecture is completely division free as the division operation is replaced by multiplication, shift and addition operations. The design was implemented on FPGA and later synthesized in CMOS 0.18 μm technology. The results show that the proposed design satisfies the requirement of all codecs with a maximum decoding capability of 60 fps at 187.3 MHz for Xilinx Virtex4 LX60 FPGA of a 1080p HD video. The scheme is also suitable for low-cost implementation in modern multi-codec systems.

## ACKNOWLEDGMENTS

The work presented in this thesis has been carried out at the Department of Electrical and Computer Engineering, University of Saskatchewan, Saskatoon, Saskatchewan during the years 2010 – 2012 under the kind supervision of Dr. Khan A. Wahid.

First I would like to express my gratitude to my supervisor, Dr. Wahid for introducing me to the research field and for his continuous support, inspiration and guideline throughout the project. His vast knowledge and expertise in this field added considerably to my graduate experience.

Then, I would like to thank my thesis committee members for their time to examine my thesis and insightful comments. I would also like to acknowledge the financial support from NSERC. Finally, I would like to thank my family for their love, encouragement and support.

- Mousumi Das

## TABLE OF CONTENTS

PERMISSION TO USE.....i

ABSTRACT.....ii

ACKNOWLEDGEMENT.....iii

TABLE OF CONTENTS.....iv

LIST OF TABLES.....vii

LIST OF FIGURES.....viii

LIST OF ABBREVIATIONS.....ix

1 INTRODUCTION.....1
- 1.1 Digital convergence in consumer products.....1
- 1.2 Different video coding standards.....2
  - 1.2.1 MPEG-2 standard.....2
  - 1.2.2 Motion JPEG (MJPEG) standard.....2
  - 1.2.3 Windows Media Video 9 (WMV-9)/VC-1 standard.....3
  - 1.2.4 H.264/MPEG-4 (part-10)/AVC standard.....4
  - 1.2.5 AVS standard.....4
  - 1.2.6 Developing HEVC standard.....5
- 1.3 Basics of video compression.....6
- 1.4 Components in a video codec.....7
- 1.5 Overview of a multi quantizer unit.....8
- 1.6 Previous works.....10
- 1.7 Thesis objectives.....12
- 1.8 Thesis organizations.....13

2 DIFFERENT QUANTIZATION SCHEMES.....14
- 2.1 Introduction.....14
- 2.2 Overview of quantization.....14
- 2.3 Quantization in H.264.....15
  - 2.3.1 8x8 quantization in H.264.....16
- 2.4 Quantization in AVS.....18
  - 2.4.1 8x8 quantization in AVS.....18
- 2.5 Quantization in VC-1.....19
- 2.6 Quantization in MPEG-2/4.....21
- 2.7 Quantization in Motion JPEG (MJPEG).....23
  - 2.7.1 8x8 quantization in MJPEG.....23

2.8 Quantization in HEVC (draft stage)................................................................24
2.8.1 Proposal by Samsung Electronics, Ltd, FastVDO...........................25
2.8.2 Proposal by Cisco Systems, Texas Instrument Inc...........................26

3 PROPOSED ALGORITHM........................................................................................27
3.1 Introduction........................................................................................27
3.2 Proposed Division Free Quantization Algorithm (DFQA)...........................27
3.2.1 Steps of DFQA..................................................................28
3.2.2 Equation to calculate MF for VC-1, MPEG-2/4 and MJPEG.........29
3.2.3 DFQA applied to H.264................................................29
3.2.3.1 Example..........................................................30
3.2.4 DFQA applied to AVS.........................................................32
3.2.4.1 Example..........................................................32
3.2.5 DFQA applied to VC-1 and MPEG-2/4.........................................33
3.2.5.1 Example..........................................................35
3.2.6 DFQA applied to MJPEG...................................................36
3.2.6.1 Example..........................................................37
3.2.7 DFQA applied to HEVC...................................................39
3.2.7.1 Example..........................................................41

4 HARDWARE IMPLEMENTATION...........................................................................42
4.1 Introduction........................................................................................42
4.2 Overall hardware architecture........................................................43
4.2.1 Architecture of the multi quantizer.......................................44
4.2.2 Architecture of the look-up tables.......................................46
4.3 Overall operation........................................................................49

5 HARDWARE COMPARISON.................................................................................52
5.1 Introduction........................................................................................52
5.2 Performance comparison in FPGA..................................................52
5.2.1 Resource utilization summary in FPGA...............................52
5.2.2 Performance comparison of the multi quantizer.......................53
5.2.3 Performance comparison of the transform-quantizer.......................55
5.3 Performance comparison in VLSI..................................................57
5.3.1 Estimated area savings in the multi quantizer.......................57
5.3.2 Estimated area savings in the multi codec decoder.......................59
5.4 Calculation and comparison of decoding capability......................................60

6 CONCLUSIONSAND SUGGESTIONS FOR FUTURE WORK....................................63
6.1 Conclusion........................................................................................63
6.2 Suggestions for future works.............................................................64

REFERENCES...........................................................................................66

# LIST OF TABLES

Table 2.1. Quantization step sizes in H.264....................17
Table 2.2. Quantization parameter and the values of MF ....................17
Table 2.3. Quantization table for AVS encoder....................19
Table 2.4. Derivation of DC StepSize for uniform and non-uniform quantizer ....................21
Table 3.1. Description of different parameters used in the DFQA....................41
Table 5.1. Resource utilization in Virtex2 FPGA....................53
Table 5.2. Resource utilization in Virtex4 FPGA....................53
Table 5.3. Resource utilization in Virtex5 FPGA....................53
Table 5.4. Comparison with the existing designs (quantizer only)....................54
Table 5.5. Comparison with the existing designs (transform and quantization)....................56
Table 5.6. Comparison of multi quant with the estimated cost (VLSI only)....................57
Table 5.7. Comparison of decoding capability of the multi quantizer schemes....................61
Table 5.8. Comparison of decoding capability of combined transform-quantizer....................61

## LIST OF FIGURES

Figure 1.1. Series of still images representing "scenes in motion"................................................6
Figure 1.2. Functional block dagram of modern video codec.......................................................8
Figure 1.3. Look-up table based multi quantizer unit.................................................................10
Figure 2.1. Four basic steps of transform coding........................................................................15
Figure 2.2. The position specification of MF..............................................................................16
Figure 2.3. VC-1 quantization and dequantization rule showing
(a) dead zone
(b) regular uniform quantization ..............................................................................20
Figure 3.1. Barbara image...........................................................................................................30
Figure 4.1. Basic quantization scheme using division operation...............................................42
Figure 4.2. Quantization scheme of the proposed multi quantizer..............................................42
Figure 4.3. Overall block diagram of the DFQA.........................................................................43
Figure 4.4. Orientation of multi quantizer, multi transform, QP processing and LUT.................44
Figure 4.5. Hardware of step 1 of the DFQA...............................................................................45
Figure 4.6. Hardware of step 2 of the DFQA...............................................................................46
Figure 4.7. Hardware of step 3 of the DFQA...............................................................................46
Figure 4.8. Look-up tables of all six standards.............................................................................47
Figure 4.9. Look-up tables of H.264 standard...............................................................................48
Figure 4.10. Look-up tables and complete data flow diagram.....................................................49
Figure 4.11. Entire operation with all pipelining boundaries.......................................................50
Figure 4.12. Timing diagram of the multi quantizer....................................................................51
Figure 5.1. Comparison of normalized area of the multi quantizer.............................................58
Figure 5.2. Cost reduction map of decoder using the multi quantizer architecture....................59

## LIST OF ABBREVIATIONS

| | |
|---|---|
| 1D | One Dimensional |
| 2D | Two Dimensional |
| ASIC | Application-Specific Integrated Circuit |
| AVC | Advance Video Codec |
| AVS | Audio Video coding Standard |
| CMOS | Complementary-symmetry Metal Oxide Semiconductor |
| DCT | Discrete Cosine Transform |
| FPGA | Field Programmable Gate Array |
| FSM | Finite State Machine |
| fps | Frame Per Second |
| HD | High Definition |
| HDL | Hardware Description Language |
| HEVC | High Efficiency Video Coding |
| IDCT | Inverse Discrete Cosine Transform |
| IEC | International Electrotechnical Commission |
| IEEE | Institute of Electrical and Electronics Engineers |
| Int-DCT | Integer Discrete Cosine Transform |
| Int-IDCT | Integer Inverse Discrete Cosine Transform |
| IP | Intra Prediction |
| IQ | Inverse Quantization |
| IT | Inverse Transform |
| ISO | International Organization for Standardization |
| ITU | International Telecommunication Union |
| JCT-VC | Joint Collaborative Team on Video Coding |
| JPEG | Joint Photographic Experts Group |
| LUT | Look-up Table |
| MC | Motion Compensation |
| MF | Multiplication Factor |
| MJPEG | Motion JPEG |
| MPEG | Moving Picture Experts Group |
| Msec | Milli-second |
| Q | Quantization |
| QP | Quantization Parameter |
| QS | Quantization Step |
| SOC | System On a Chip |
| UHD | Ultra High Definition |
| VC-1 | Video Codec -1 |
| VCEG | Video Coding Experts Group |

| | |
|---|---|
| VLD | Variable Length Decoder |
| VLSI | Very Large Scale Integration |
| WMV9 | Windows Media Video 9 |

CHAPTER 1

# INTRODUCTION

This chapter presents a brief description of materials which are necessary for this research work. It begins with the current demand of the modern world focusing on digital convergence of electronic consumer products in a single system-on-chip (SoC) platform. Then, different video coding standards are described briefly followed by basics of video compression. After that, the units of modern video codecs are discussed. This initiates the need of the multi quantizer unit which is the main focus of this thesis. Finally, previous research works have been presented.

## 1.1 Digital convergence in consumer products

An evident trend in modern world is the digital convergence in the current electronic consumer products. Recently, reconfigurable concept has become one of the most important issues for video coding technology. Nowadays people usually want a product that can support various formats and features, such as, Video on Demand (VOD), Digital Multimedia Broadcasting (DMB) and Portable Multimedia Player (PMP) and so on. Due to such demand and realizing the fact that the intercommunication among different video devices supporting different standards is inconvenient, it is necessary to support widely used video compression standards in a single system-on-chip (SoC) platform. So the goal is to find a way to develop a multi-codec system that achieves high performance i.e. higher decoding capability, as well as low hardware cost. To develop reconfigurable architecture or algorithm which meets both of the objectives is a challenge.

With the significant development of multimedia technology during the last decade, various video coding standards, such as, MJPEG [1], MPEG-2/4 [2], VC-1 [3], H.264/AVC [4], AVS [5] has been developed for different applications. Moreover, to improve the coding efficiency further, a joint collaborative team on video coding (JCT-VC) has starred drafting a new video standard known as High Efficiency Video Coding (HEVC) [6]. Therefore, a multi-codec system that supports all these standards is needed to satisfy the requirements of different applications.

## 1.2 Different video coding standards

There are several video coding standards in use. The following sections describe six popular video coding standards.

### 1.2.1 MPEG-2 standard

MPEG-2 is a standard for generic coding of moving pictures and associated audio information [7]. MPEG-2 is widely used as the format of digital television signals that are broadcasted by terrestrial, cable and direct broadcast satellite TV systems. It also specifies the format of movies and other programs that are distributed on DVDs and similar discs. All standards-compliant MPEG-2 video decoders are fully capable of playing back MPEG-1 Video streams conforming to the Constrained Parameters Bitstream syntax. With some enhancements, MPEG-2 Video and Systems are also used in some HDTV transmission systems.

### 1.2.2 Motion JPEG (MJPEG) standard

Motion JPEG (MJPEG) standard separately compresses each frame of video sequence in JPEG format [8]. Hence this standard does not exploit the redundancy presented in successive frames. As a result as compared to MPEG, MJPEG has lower attainable compression but is also less computationally complex. It is popularly used for non-linear editing which requires easy access to any frame. Another application for MJPEG is medical imaging which requires high quality images and error resilience. The disadvantage of MJPEG is that it does not exploit the temporal redundancies of successive frames to achieve higher compression and consequently MJPEG has higher bit-rate than MPEG for the same quality. Typical quality problems for MJPEG are blocking and ringing artifacts, especially at low bit-rate compression.

### 1.2.3 Windows Media Video 9 (WMV-9)/VC-1 standard

Microsoft Windows Media 9 series is a set of technologies that enables rich digital media experiences across many types of networks and devices [9]. These technologies are widely used in industry for media delivery over the internet and other media, and are also applied to broadcast, high definition DVDs and digital projection in theaters. At the core of these technologies is a state-of-the-art video codec called Windows Media Video 9 (WMV-9), which provides highly competitive video quality for reasonable computational complexity. Although the origins of Windows Media focused on streaming compressed audio and video over the internet to personal computers, the vision moving forward is to enable effective delivery of digital media through any network to any device. In addition to Internet-based applications (e.g., subscription services, video on demand over IP, web broadcast, etc.), content compressed with Windows Media codecs is being consumed by a wide range of wired and wireless consumer electronic devices such as mobile phones, DVD players, portable music players, car stereos, etc.

### 1.2.4 H.264/MPEG-4 (part-10)/AVC standard

The continuing development of digital video coding has produced ITU-T H.264/MPEG-4 (part-10), also known as Advanced Video Coding (AVC)[4]. H.264/MPEG-4 (part-10)/AVC is a block-oriented motion-compensation-based codec standard developed by the ITU-T Video Coding Experts Group (VCEG) together with the International Organization for Standardization (ISO)/International Electro technical Commission (IEC) joint working group, the Moving Picture Experts Group (MPEG). It provides gains in compression efficiency up to 50% over the wide range of bit rates and video resolution compared to the previous video standards [10]. Compared to the other video standards, H.264 has many features that make it one of the most powerful standards. Besides, network friendliness and good video quality at high and low bit rates are important features that distinguish H.264 from other standards. This video compression standard is currently one of the most commonly used formats for the recording, compression, and distribution of high definition video. Moreover, H.264 video format has a very broad application range that covers all forms of digital compressed video from low bit-rate internet streaming applications to HDTV broadcast and Digital Cinema applications with nearly lossless coding.

### 1.2.5 AVS standard

AVS (Audio Video Coding Standard), developed by the China AudioVideo Coding Standard Working Group, was adopted by the Chinese government to reduce foreign dependence on core intellectual properties used in digital media technology [11]. Proposed as a national standard in 2004, AVS soon became the mandatory video compression standard in China terrestrial digital TV. AVS-video is an application driven coding standard. AVS Part 2 targets high-definition digital video broadcasting and high-density storage media and AVS Part 7 targets low complexity, low picture resolution mobility applications [11]. Integer transform, intra and inter-picture prediction, in-loop deblocking filter and context-based two dimensional variable length coding are the major components in AVS-video compression, which are well-tuned for target applications. It achieves similar performance to H.264/AVC with lower cost. Up to now, there are two separate parts in this standard targeting to different video compression applications: (a) AVS Part 2 for high-definition digital video broadcasting and high-density storage media; (b) AVS Part 7 for low complexity, low picture resolution mobility applications. The informal names, AVS 1 .0 and AVS-M, are used to represent for AVS Part 2 and Part 7 respectively [11].

### 1.2.6 Developing HEVC standard

High Efficiency Video Coding (HEVC) is a draft video compression standard, a successor to H.264/MPEG-4 AVC (Advanced Video Coding) currently under joint development by the ISO/IEC Moving Picture Experts Group (MPEG) and ITU-T Video Coding Experts Group (VCEG) [6]. HEVC is said to improve video quality, double the data compression ratio compared to H.264, and can support resolutions up to 8K UHD (7680 × 4320). It aims to substantially improve coding efficiency compared to AVC High Profile, i.e., to reduce bitrate requirements by half with comparable image quality, at the expense of increased computational complexity. Two of the key features where HEVC was improved compared to H.264/MPEG-4 AVC were support for higher resolution video and improved parallel processing methods. HEVC is targeted at next-generation HDTV displays and content capture systems which feature progressive scanned frame rates and display resolutions from QVGA (320×240) up to 1080p (1920×1080) and 4320p (7680×4320), as well as improved picture quality in terms of noise level, color gamut, and dynamic range.

## 1.3 Basics of video compression

Video refers to pictorial (visual) information, including still images and time-varying images. A series (or set) of still images which represents "scenes in motion" is referred to as video which is shown by Figure 1.1. A time varying image is a spatiotemporal intensity pattern, denoted by $s_c(x, y, t)$, where $x$ and $y$ is the spatial variable and $t$ is the temporal variable.

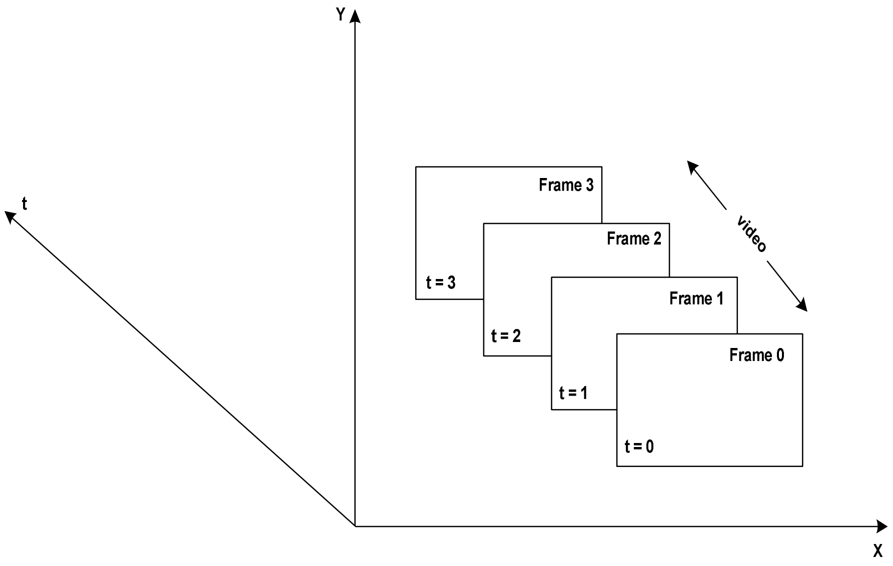

Figure 1.1. Series of still images representing "scenes in motion"

A typical video scene is composed of multiple objects each with their own characteristics such as shape, depth, texture, and illumination. An object has two characteristics: (i) spatial characteristics (texture variation, number and shape of objects, color, etc) and (ii) temporal characteristics (object motion, changes in illumination (brightness), movements of camera, etc). However, video compression is necessary everywhere in our modern day life as it helps reduce resource usage, such as data storage space or transmission capacity. Since a video stream has two spatial and one temporal dimension, video compression is usually done independently in all dimensions. A common characteristic of an image is that the neighboring pixels are correlated and therefore, contains redundant information. So, image compression is achieved by eliminating redundancies of neighboring pixels and frames. There are three types of redundancies to consider. (i) spatial redundancies or correlation between neighboring pixel values (ii) spectral redundancies or correlation between different color planes or spectral bands (iii) temporal redundancy or correlation between adjacent frames in a sequence of images (in video applications). In video compression, the encoder removes spatial and temporal redundancies and the decoder puts them back. However, in modern video compression technique exploiting the spatial and spectral redundancies is not enough. So the modern multimedia codecs employ another technique known as, motion compensation/estimation which is the temporal redundancy.

## 1.4 Components in a video codec

Most modern multimedia codecs (both encoder and decoder) employ transform-quantization pair as well as entropy coding, motion compensation/estimation as shown in Figure 1.2. The transform-quantization pair is very important for lossy video compression. The transform unit isolates various image frequencies, i.e. low frequency and high frequency components. Low frequencies correspond to the important image features, whereas high frequencies correspond to the details of the image, which are less important as human visual system is less sensitive to high frequency components. The transform coefficients go through the quantization stage to reduce perceptual redundancies and bit rates. A quantization parameter at the encoder divides transform coefficients, and the quotient is transmitted. The decoder multiplies this quotient by the same quantization parameter to reconstruct the coefficient. Thus the quantization unit quantizes each transform coefficient separately coming from the transform unit. During this process, the pixels that correspond to high frequencies can be quantized heavily, whereas pixels that correspond to low frequencies can be quantized lightly or not at all. This is how a transform-quantization pair can compress an image very effectively by losing information, associated with unimportant image details. The encoder treats the quantized coefficients as symbols which are then entropy encoded. The motion estimation stage of an encoder finds matching blocks in reference pictures that are similar to the block that's currently being coded. To compensate for motion in a video, the matched block is subtracted from the current block. The closer the block match, the smaller the difference, and the smaller the amount of information transmitted to a decoder.

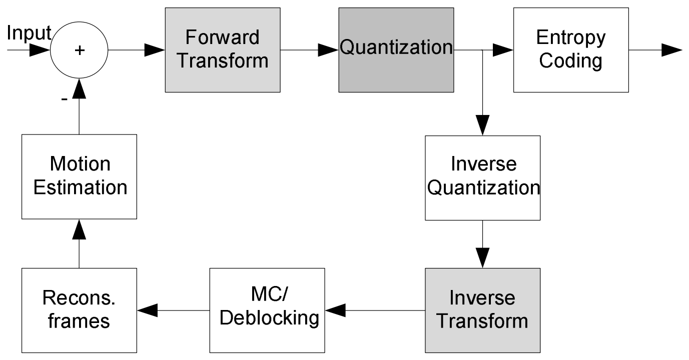

Figure 1.2. Functional block diagram of modern video codec

The quantization unit serves a key role in video compression and the current demand leads to various devices that support different features, therefore, the focus is on developing and efficiently implementing a multi-quantizer unit to compute quantization of six widely used multimedia codecs: MJPEG, MPEG-2/4, VC-1, H.264/AVC, AVS and HEVC.

## 1.5 Overview of a multi quantizer unit

Different video standard has different quantization parameters and quantization matrices. While developing the architecture, all the quantization (Q) coefficients of the Q-tables of different standards have been considered carefully to establish a relationship between them. The quantization in Motion JPEG (MJPEG) is defined as the division of the DCT coefficient by the corresponding Q-values specified by the Q-matrices. Similarly for MPEG-2, according to [7] the quantized value is obtained by dividing the DCT coefficient by the corresponding values of the Q matrices. The work in [12] states that MPEG-4 part 2 supports custom quantization table that can be tuned to content and bit rate like MPEG-2. Hence in this proposed design the same quantization scheme for both MPEG-2 and MPEG-4 part 2 is used and referred to as the quantization of MPEG-2/4. On the other hand, the two most popular video standards, H.264/AVC and AVS, exploit multiplication and shift operation for the purpose of quantization to avoid the division operation for reduced computational complexity. Moreover, HEVC uses similar Q-scheme as H.264. The quantization in VC-1 is user-defined and similar to the process in MPEG-2 [9].

Based on the observation, a new multi-quantizer architecture to support these six codecs is developed. This architecture, shown in Figure 1.3 follows a look-up table based approach. It is completely division free, as division operation is replaced by multiplication, addition and shift operation. Note that the right shift operation can be a replacement of division operation. For example, the binary of 16 is $(10000)_2$. Now if 1 is right shifted by one bit, the result is $(01000)_2$ which is the binary form of 8. So, right shift is equivalent to the division operation. In addition, the right shift operation reduces the hardware complexity as compared to the division. The multi quantizer architecture has only one shared multiplier and one control logic for all the six standards. The concept of using shared multiplier and control logic reduces the hardware cost as well as less computational complexity.  The architecture is later synthesized into both FPGA and ASIC level and the result is compared with existing designs. The proposed design serves as a key unit in a multi-codec system in transcoding applications [13] and [14].

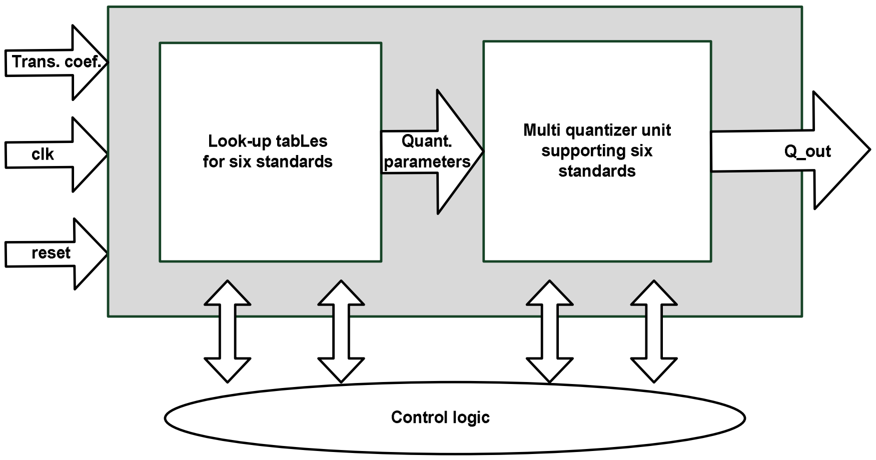
Trans. coef.
clk
reset
Look-up tabLes for six standards
Quant. parameters
Multi quantizer unit supporting six standards
Q_out
Control logic

Figure 1.3. Look-up table based multi quantizer unit

## 1.6 Previous works

A significant amount of research has been conducted to efficiently combine and implement the transform units for multiple codecs [15-18]. On the other hand little research is focused on the implementation of multi-quantizer unit. Among the multiple-transform units, a unified Inverse Discrete Cosine Transform (IDCT) architecture to support five standards (such as, AVS, H.264, VC-1, MPEG-2/4 and JPEG) is presented in [15]. The authors in [16] offer an area efficient architecture to perform a DCT-based transform for JPEG, MPEG-4, VC-1 and H.264 using delta mapping. The design in [17] is an IDCT and IQ circuit for H.264, MPEG-4 and VC-1. However, it shows a general implementation of the inverse quantization. The full IQ design including all the quantization tables of these three standards is not implemented. The authors in [18] present a design to support 8x8 transform and quantization for H.264, MPEG-4 and VC-1 where the quantization circuit is again a generalized implementation like the design in [17]. A design to support the 4x4 transform and quantization of H.264 has been presented in [19]. The 8x8 transform and quantization for H.264 is presented in [20] and [21]. Several other designs based on H.264 codec have been reported in [22-27]. The authors in [28] present a design for the quantization for AVS. The design in [29] describes an MPEG-2 encoder. In [30], another JPEG encoder is implemented for images where the quantization block is designed using multiplication and shift operation instead of division. The design in [31] describes a multi standard video decoder to support four codecs - AVS, H.264, VC-1 and MPEG-2. Silicon Image Inc. currently supplies a Multi-standard High-Definition Video Decoder (MSVD-HD) core that supports H.264, VC-1, and MPEG- 1/2 codecs [32]. Their multiplexed decoder chip costs 970K gates using TSMC 90 nm technology (including complete memory interfacing, stream reader functionality and extra logic for context switch support). In a recent JCT-VC meeting, the transform and quantization algorithms for the new developing standard, HEVC are finalized [33]. The work in [34] presents a multi-codec design for IDCT implementation including HEVC. However, none of the existing designs can compute the quantization of all six video codecs. In this thesis, a new division-free quantization algorithm (DFQA) is presented to compute the quantization units for six multimedia codecs: MJPEG, MPEG-2/4, VC-1, H.264/AVC, AVS and HEVC. In addition, the dedicated hardware implementation as compared to a software program executed on a general purpose processor has several advantages [35]. The hardware implementation is necessary to increase the speed and efficiency of operation of the whole process. The software simulation is executed sequentially [36]. On the other hand, complete parallel architecture can be realized in hardware based implementation. Moreover, it is well known that the dedicated FPGA based processor offer high data rate over the general purpose software processors. So in high data throughput application such as video processing applications, FPGAs provide throughput advantages over the general purpose processors due to the massive parallelism capability they offer. The authors of [35] present an assessment of hardware vs. software implementations for video microscopy. The results show that the hardware based implementations speedup up to ~5 times as compared to the software based implementation. Due to all these advantages, an efficient hardware implementation of division free multi quantizer architecture has been presented in this thesis.

## 1.7 Thesis objectives

The objectives of this thesis is to develop a unified architecture on a single chip that performs 8x8 quantization operation for the six modern video coding standards, (AVS, H.264, VC-1, MPEG-2/4, MJPEG and HEVC). The focus is on the efficient implementation of multiple quantization units; the implementation of the full encoder (or decoder) is not explored. However, integrating multiple quantization units of different standards into a single chip increases the area (size) and decreases the frequency, which has negative impact on the overall performance. Therefore, the ultimate goal is to design and implement a low cost multi quantizer unit that meets the real-time performance requirements. To accomplish this goal, the thesis work is focused on following points:

1. Investigate different quantization schemes and parameters to find a relationship among them and consequently develop a generalized quantization algorithm which can be applied to all the six standards.

2. Implement the developed algorithm in hardware using Verilog HDL. Circuit sharing strategy is explored to reduce the hardware cost and computational complexity.

3. Verify the functionality and synthesize the developed design in both FPGA (Field Programmable Gate Array) and ASIC (Application-Specific Integrated Circuit).

4. Compare the performance of the design with that of the existing designs in terms of hardware count, operational frequency and decoding capability.

## 1.8 Thesis organization

The thesis is organized into six chapters. Chapter 1 gives an overview of need of multi codec system followed by brief description of different video coding standards. It also illustrates the basics of video compression and modern multimedia codec system which in turn presents the need of a multi quantizer unit which is the main part of this dissertation. Objectives of this thesis dissertation are also described in this chapter.

Chapter 2 presents brief description of quantization operation of six different standards. It includes different quantization equations for different standards, different quantization matrices and other quantization parameters specified by these standards.

In Chapter 3 the generalized Division Free Quantization Algorithm (DFQA) has been described.

Chapter 4 illustrates the entire circuit shared hardware architecture which includes the multi quantizer unit, arrangement of Look-up tables and the control logic.

The performance comparison is presented in Chapter 5. Finally, Chapter 6 presents conclusion and future works.

CHAPTER 2

# DIFFERENT QUANTIZATION SCHEMES

## 2.1 Introduction

This chapter presents an overview of quantization schemes of six different video codecs-H.264/AVC, VC-1, AVS, MPEG-2/4, MJPEG and HEVC (under development). Here different quantization parameters, quantization matrices, quantization equations etc. has been described briefly for all the standards.

## 2.2 Overview of quantization

Most modern multimedia codecs (both encoder and decoder) employ transform-quantization pair. Transform coding is the basis of all lossy compression which has four basic steps shown by Figure 2.1. A basic transform coder segments the image into smaller square blocks. Each block undergoes a 2-D orthogonal transformation. The transform coefficients are individually quantized and coded (the coefficients with higher energy are finely quantized). The encoder treats the quantized coefficients as symbols which are then entropy encoded. Low frequency components in image correspond to important image features whereas the high frequency components correspond to the details of the image which are less important. The transform isolates various image frequencies. Thus the pixels correspond to high frequencies are quantized heavily. On the other hand, the pixels correspond to lower frequencies are quantized lightly. In some cases, they are not quantized at all.

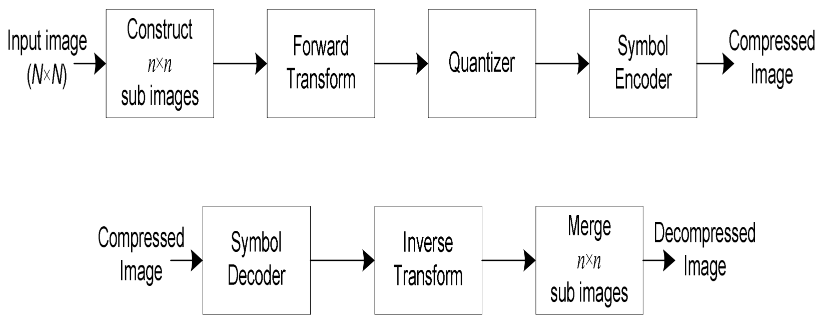

Figure 2.1. Four basic steps of transform coding

Different video coding standard has different quantization parameters and algorithm specified by that particular standard. The following sections demonstrate a brief description of quantization of different standards.

## 2.3 Quantization in H.264

A key approach to transform and quantization in H.264 is to intentionally magnify the transform in the middle (just before division part of quantization) to involve the right shift operations in both encoder and decoder. This right shift operation is a part of quantization process. Typically quantization loses some resolution in the encoder due to the division operation, thus amplifying the random noise in the decoder due to multiplication of the inverse quantization. In contrast, only right shift operation is performed in both encoder and decoder in H.264 – with no random noise amplified during the decoder's inverse quantization stage. In H.264, the overall quantization is performed by multiplication and right shift operation. This standard defines its own multiplication factor (MF). These MFs are multiplied with the transform coefficients and finally right shifted to obtain the final quantized value. As a result, the quantization in H.264 does not need any direct division operation.

### 2.3.1 8x8 quantization in H.264

H.264 standard supports both 4x4 and 8x8 quantization. The 8x8 quantization with all the quantization parameter is briefly discussed below. For a given step size, the encoder can perform quantization by the following equation:

$$y = (w[i,j] \,.\, MF[i,j] + f << qbits) >> qbits, \text{for } i,j = 0,\ldots,7; \qquad (2-1)$$

Where, $w$ denotes the transform coefficient, $i$ and $j$ are the row and column indices, $qbits = (16 + QP/6)$, $MF$ is the Multiplying factor and $y$ denotes the corresponding quantized value (level). $QP$ is the Quantization Parameter which specifies $MF$s, $f$ is the offset value ranging from 0 to 0.5 and is chosen by the encoder. Here, $QP$ is the quantization parameter ranging from 0 to 51. In case of H.264 as specified in, Multiplication Factor $MF$ depends on m (= *QP/*6) and the position $(i,j)$ of the element is given by Figure 2.2.

$$MF[m;i,j] = \begin{cases} M_{m0} & \text{for} \quad (i,j) \text{ with} \quad i=[0,4], j=[0,4] \\ M_{m1} & \text{for} \quad (i,j) \text{ with} \quad i=[1,3,5,7], j=[1,3,5,7] \\ M_{m2} & \text{for} \quad (i,j) \text{ with} \quad i=[2,6], j=[2,6] \\ M_{m3} & \text{for} \quad (i,j) \text{ with} \quad (i=[0,4], j=[1,3,5,7]) \cap (i=[1,3,5,7], j=[0,4]) \\ M_{m4} & \text{for} \quad (i,j) \text{ with} \quad (i=[0,4], j=[2,6]) \cap (i=[2,6], j=[0,4]) \\ M_{m5} & \text{for} \quad (i,j) \text{ with} \quad (i=[2,6], j=[1,3,5,7]) \cap (i=[1,3,5,7], j=[2,6]) \end{cases}$$

Figure 2.2. The position specification of *MF*

According to [7], total of 52 values of $Qstep$ are supported by the standard and these are indexed by $QP$. The values of $Qstep$ corresponding to each $QP$ are shown in Table 2.1. Note that, $Qstep$ doubles in size for every increment of 6 in $QP$ and $Qstep$ increases by 12.5% for each increment of 1 in $QP$.

Table 2.1. Quantization step sizes in H.264

| *QP* | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Qstep* | 0.625 | 0.6875 | ... | ... | 1 | 1.125 | ... | ... | 1.625 | ... | 2 | ... | 2.5 |
| *QP* | … | 18 | … | 24 | … | 30 | … | 36 | … | 42 | … | ... | 51 |
| *Qstep* | … | 5 | … | 10 | … | 20 | … | 40 | … | 80 | … | ... | 224 |

This standard specifies the first six values of $MF$ as shown in Table 2.2. For $QP > 5$ the factor $MF$ remains unchanged but the value of $qbits$ will change. For example, $qbits = 16$ for $0 \le QP \le 5$ (as $qbits = 16 + QP/6$), $qbits = 17$ for $6 \le QP \le 11$ and so on.

Table 2.2. Quantization parameter and the values of MF

| *QP* | *MF* | | | | | |
|---|---|---|---|---|---|---|
| 0 | 13107 | 11428 | 20972 | 12222 | 16777 | 15481 |
| 1 | 11916 | 10826 | 19174 | 11058 | 14980 | 14290 |
| 2 | 10082 | 8943 | 15978 | 9675 | 12710 | 11985 |
| 3 | 9362 | 8228 | 14913 | 8931 | 11984 | 11259 |
| 4 | 8192 | 7346 | 13159 | 7740 | 10486 | 9777 |
| 5 | 7282 | 6428 | 11570 | 6830 | 9118 | 8640 |

Table 2.2 is used to generate six 8x8 matrices corresponding to each $QP$ in accordance with the position defined in Figure 2.2. For $QP > 5$ same matrices will be used. For example, for $QP = 6$ the same matrix for $QP = 0$ will be used. Similarly for $QP = 7$ the same matrix for $QP = 1$ will be used and so on. Equation (2-2) shows the 8x8 matrix for $QP = 0$. Other matrices can be developed in the same way.

$$MF\_0 = \begin{bmatrix} 13107 & 12222 & 16777 & 12222 & 13107 & 12222 & 16777 & 12222 \\ 12222 & 11428 & 15481 & 11428 & 12222 & 11428 & 15481 & 11428 \\ 16777 & 15481 & 20972 & 15481 & 16777 & 15481 & 20972 & 15481 \\ 12222 & 11428 & 15481 & 11428 & 12222 & 11428 & 15481 & 11428 \\ 13107 & 12222 & 16777 & 12222 & 13107 & 12222 & 16777 & 12222 \\ 12222 & 11428 & 15481 & 11428 & 12222 & 11428 & 15481 & 11428 \\ 16777 & 15481 & 20972 & 15481 & 16777 & 15481 & 20972 & 15481 \\ 12222 & 11428 & 15481 & 11428 & 12222 & 11428 & 15481 & 11428 \end{bmatrix} \qquad (2-2)$$

## 2.4 Quantization in AVS

Transform and quantization in AVS is based on 4x4 and 8x8 block. The forward quantization that quantizes the transform coefficients are mainly consists of two steps, (i) scaling and (ii) quantization. The 4x4 and 8x8 quantization scheme has two different 4x4 and 8x8 scale matrices. The structure of 4x4 and 8x8 quantization equations is similar with slightly different quantization parameters.

### 2.4.1 8x8 quantization in AVS

The quantization scheme of AVS standard undergoes through Scaling process followed by the quantization operation on the scaled coefficients. The scaling process to the transform coefficients are defined by equation (2-3) [28]:

$$d = (w[i,j]\,.\,scaleM + 1 << 18) >> 19, \text{ for } i,j = 0,\ldots,7; \qquad (2-3)$$

The quantization is performed to the scaled coefficient obtained in first step by the equation (2-4). The scale matrix based on [28] is illustrated by (2-5). Value of scale parameter varies for different coefficient position in 8x8 matrix.

$$y = (d[i,j]\,.\,MF[QP] + 1 << qbits) >> (1 + qbits), \text{ for } i,j = 0,\ldots,7; \qquad (2-4)$$

Where, $w$ denotes the transform coefficient, $i$ and $j$ are the row and column indices, $qbits = 14$, $MF$ is the Multiplying factor which depends on $QP$ and $y$ denotes the corresponding quantized value (level). $QP$ is the Quantization Parameter ranging from 0 to 63. According to [28] each $QP$ specifies one particular $MF$ described by Table 2.3.

$$ScaleM = \begin{bmatrix} 32768 & 37958 & 36158 & 37958 & 32768 & 37958 & 36158 & 37958 \\ 37958 & 43969 & 41884 & 43969 & 37958 & 43969 & 41884 & 43969 \\ 36158 & 41884 & 39898 & 41884 & 36158 & 41884 & 39898 & 41884 \\ 37958 & 43969 & 41884 & 43969 & 37958 & 43969 & 41884 & 43969 \\ 32768 & 37958 & 36158 & 37958 & 32768 & 37958 & 36158 & 37958 \\ 37958 & 43969 & 41884 & 43969 & 37958 & 43969 & 41884 & 43969 \\ 36158 & 41884 & 39898 & 41884 & 36158 & 41884 & 39898 & 41884 \\ 37958 & 43969 & 41884 & 43969 & 37958 & 43969 & 41884 & 43969 \end{bmatrix} \quad (2-5)$$

Table 2.3. Quantization table for AVS encoder

| *QP* | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| *MF* | 32768 | 29775 | 27554 | 25268 | 23170 | 21247 | 19369 | 17770 |
| *QP* | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| *MF* | 16302 | 15024 | 13777 | 12634 | 11626 | 10624 | 9742 | 8958 |
| *QP* | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 |
| *MF* | 8192 | 7512 | 6889 | 6305 | 5793 | 5303 | 4878 | 4467 |
| *QP* | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| *MF* | 4091 | 3756 | 3444 | 3161 | 2894 | 2654 | 2435 | 2235 |
| *QP* | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 |
| *MF* | 2048 | 1878 | 1722 | 1579 | 1449 | 1329 | 1218 | 1117 |
| *QP* | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 |
| *MF* | 1024 | 939 | 861 | 790 | 724 | 664 | 609 | 558 |
| *QP* | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 |
| *MF* | 512 | 470 | 430 | 395 | 362 | 332 | 304 | 279 |
| *QP* | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 |
| *MF* | 256 | 235 | 215 | 197 | 181 | 166 | 152 | 140 |

## 2.5 Quantization in VC-1

VC-1 uses multiple transform sizes but the same quantization rule is applied to all the coefficients. This standard allows both dead-zone and regular uniform quantization. The regular uniform quantization is the simplest but most pervasive method to quantize DCT transform coefficient. In uniform quantization the quantization intervals are identical. The dead zone quantization uses similar principle as the uniform quantization but it has a larger interval, i.e. larger quantization step size than the uniform quantization around zero which is called the "dead zone". All the coefficients falling in this interval (small coefficients close to zero) are quantized to zero. All the quantization intervals except the dead-zone are of same size - the dead-zone being typically larger. The use of dead-zone leads to substantially bit savings at low bitrates. The dead-zone and regular uniform quantization with the reconstruction level is shown in Figure 2.3 (a) and (b).

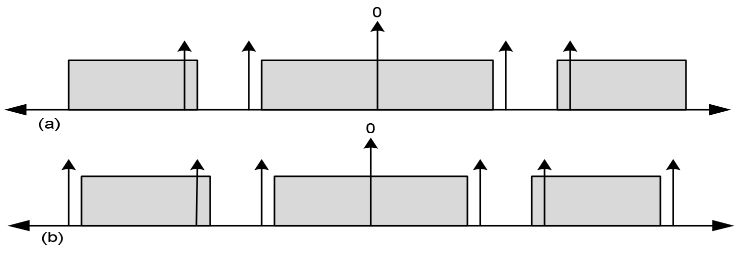

Figure 2.3. VC-1 quantization and dequantization rule showing (a) dead-zone and (b) regular uniform quantization-arrows are reconstructed levels and gray boxes are recommended quantization bins (for alternate intervals)

The reconstruction levels obtained from the regular uniform dequantization are equally spaced. In practice, a dead-zone may be applied on the encoder side, but this is not revealed by the reconstruction set. As this standard allows both dead-zone and regular uniform quantization, the specific type of quantization is signaled at the frame level and appropriate dequantization rule is applied to all coefficients within the frame by the decoder.

There are two inverse quantization methods, (i) uniform quantizer and (ii) non-uniform quantizer [7]. The uniform quantizer method is composed of two sub cases – one for intra DCs and the other for all other ACs. The intra DC is constructed as $DC = qDC * DCStepSize$, where $qDC$ is a quantized DC differential value and $DCStepSize$ is shown by Table 2.4.

Table 2.4. Derivation of DC StepSize for uniform and non-uniform quantizer

| Block Type | MQUANT ≤ 2 | 3 ≤ MQUANT ≤ 4 | 5 ≤ MQUANT |
|---|---|---|---|
| Luma | 2 × MQUANT | 8 | MQUANT/2 + 6 |
| Chroma | 2 × MQUANT | 8 | MQUANT/2 + 6 |

All other ACs are reconstructed as $F'' = qF * (2 * MQUANT + \vartriangle(syntax))$; where $\vartriangle(syntax) = \text{HALFQP}$ when PQUANT based decoding is performed, whereas $\vartriangle(syntax) = 0$ when VOPDQUANT based decoding is performed.

Second, the non-uniform based quantization is performed in two sub cases, one for intra DC and the other for all other ACs. The intra DC in non-uniform quantizer is reconstructed the same way as that of the uniform quantizer. All other ACs are reconstructed as $F'' = qF * (2 * MQUANT + \vartriangle(syntax)) + sign(qF) * MQUANT$. The coefficients obtained from the IQ arithmetic are rounded to the nearest integer. The quantization for both uniform and non-uniform quantizer is just the inverse operation of the IQ arithmetic.

However the quantization in VC-1 is user defined. At a high level the quantization process (scalar quantization where each transform coefficient is independently quantized and coded) in VC-1 is similar to the corresponding process in MPEG-2 standard. So they can use the same quantization scheme.

## 2.6 Quantization in MPEG-2/4

The MPEG-2 video standard defines only the inverse quantization (IQ) process at the decoder [7]. Therefore the quantization (Q) process in the encoder can be inferred from the IQ definition of the decoder. The work in [12] states that MPEG-4 part 2 supports custom quantization table that can be tuned to content and bit rate like MPEG-2 standard. Hence they can use identical quantization approach.

The quantizer method is composed of two sub cases – one for intra DCs and the other for all other ACs [7]. The intra DC is constructed as $DC = qDC * qs$, where $qDC$ is a quantized DC differential value and $qs$ is the quantization step. The intra AC is reconstructed as $F" = (2 * qF * qs * qm) / 32$, where $qF$ is the quantized AC value, $qs$ is the quantization step and $qm$ is the quantization matrix parameter. Hence the quantization process for the intra DC is performed as $qDC = DC / qs$ and that for intra AC is performed as $qF = (16 * w) / (qm * qs)$, where $w$ is the transform coefficient.

The non-intra DC is reconstructed the same way as the intra DC. However the non-intra AC is reconstructed as $F" = ((2 * w + sign(qF)) * qs * non\_qm) / 32$, where $non\_qm$ is the non-intra quantization matrix parameter. Hence the quantization process for the non-intra DC and AC can be derived in the similar way as the intra DC and AC. MPEG-2 defines the intra and non-intra quantization matrices as specified by (2-6) and (2-7).

$$\text{int}ra\_Q = \begin{bmatrix} 8 & 16 & 19 & 22 & 26 & 27 & 29 & 34 \\ 16 & 16 & 22 & 24 & 27 & 29 & 34 & 37 \\ 19 & 22 & 26 & 27 & 29 & 34 & 34 & 38 \\ 22 & 22 & 26 & 27 & 29 & 34 & 37 & 40 \\ 22 & 26 & 27 & 29 & 32 & 35 & 40 & 48 \\ 26 & 27 & 29 & 32 & 35 & 40 & 48 & 58 \\ 26 & 27 & 29 & 34 & 38 & 46 & 56 & 69 \\ 27 & 29 & 35 & 38 & 46 & 56 & 69 & 83 \end{bmatrix} \qquad (2-6)$$

$$non\_intra\_Q = \begin{bmatrix} 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \end{bmatrix} \qquad (2-7)$$

## 2.7 Quantization in Motion JPEG (MJPEG)

The MJPEG standard defines quantization as the division operation of the DCT coefficient coming from the transform unit by the corresponding Q value (specified by the quantization matrix). MJPEG allows specification of Q-matrices that facilitates the allocation of more bits for the representation of coefficients which are visually more significant.

### 2.7.1 8x8 quantization in MJPEG

Although this standard defines quantization as the division of the DCT by the corresponding quantization matrix coefficients, there is no specific quantization matrix defined by this standard. It is progressive of the user to select a quantization matrix. However there are two default quantization matrices specified by [37] for Luma and Chroma. The elements of these matrices are based on the visibility of individual 8x8 DCT basis functions with a viewing distance equal to six times the screen width. The Luminance and chrominance quantization matrices specified by [37] are given by (2-8) and (2-9).

$$Lum\_Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix} \qquad (2-8)$$

$$chrom\_Q = \begin{bmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{bmatrix} \qquad (2-9)$$

The quantization is performed as $qF = \text{round}(F(u,v)/qm(u,v))$, where $F(u,v)$ is the transformed coefficient and $Q(u,v)$ is the corresponding matrix coefficient. This standard also defines Quality Factor (QF) to vary the image quality. Quality Factor is a number ranging from 1 to 100. The lower the Quality Factor the more the compression. A Quality Factor of 100 does not mean lossless compression instead it generally represents the Quality Factor that will generate the highest quality compressed image for the scaling algorithm. The work in [38] shows how the varying quality factor changes the JPEG image quality.

## 2.8 Quantization in HEVC (draft stage)

High Efficiency Video Coding (HEVC), successor to H.264/MPEG-4 AVC, is a new standard under development that aims to substantially improve coding efficiency compared to AVC High Profile. The development of this standard has been undertaken by a new Joint Collaborative Team on Video Coding (JCT-VC) formed by the two organizations, ITU-T VCEG and ISO/IEC MPEG. The HEVC standard is intended to improve significantly better compression capability than the existing H.264/AVC (ITU-T H.264 ISO/IEC MPEG-4 part-10) standard. The quantization in HEVC does not use the division operation rather it uses multiplication and shift operation to replace the division operation similar to H.264 standard. This developing standard supports several transform sizes ranging from 4x4 to 32x32. However the quantization/dequantization scheme is same for all the transform sizes where the multiplier depends on value of the Quantization Parameter and the shifts depend on the transform size. As the standard has not yet been finalized, some quantization algorithm proposals which are consistent to each other are described below:

### 2.8.1 Proposal by Samsung Electronics, Ltd, FastVDO

According to [39], the quantization is implemented by the equation (2-10):

$$qF = \left(w*MF + offset\right) >> \left(21 + QP/6 - M - \left(B-8\right)\right) \qquad (2-10)$$

Where, $offset = (1 << (M - 2 + (B-8)))$ and $B$ =internal bit depth (8 or 10 bit), $M = \log2\left(N\right)$, $N$ = transform size, $MF = f\left(QP/6\right)$. $QP$ is the quantization parameter ranging from 0 to 51. The values of $MF$ are given below:

$$MF = \{26214, 23302, 20560, 18396, 16384, 14564\} \qquad (2-11)$$

For $QP$ = 0 to 5, $QP$/6=0 and hence the corresponding $MF = 26214$. Similarly for $QP/6 = 5$ , corresponding $MF = 14564$.

Dequantizer equation is specified as follows:

$q = ((qF * MF' << (QP/6)) + offset) >> (M - 1(B-8))$, where $offset$ is same as the quantizer and $MF' = g\left(QP/6\right)$ which is specified as follows:

$$MF' = \{40, 45, 51, 57, 64, 72\} \qquad (2-12)$$

### 2.8.2 Proposal by Cisco Systems, Texas Instrument Inc

The quantization is implemented by the following equation based on [33] which is similar to the proposal of Samsung Electronics discussed earlier:

$$qF = \left(w * MF + offset\right) >> \left(21 + QP/6 - M - \left(B - 8\right)\right) \qquad (2-13)$$

Where $offset = (1 << (M-2) + (B-8))$, $B$ =internal bit depth (8 or 10 bit), $M = \log2\left(N\right)$, $N$ = transform size, $MF = f\left(QP/6\right)$. $QP$ is the quantization parameter ranging from 0 to 51. The values of $MF$ are given below:

$$MF = \{26214, 23302, 20560, 18396, 16384, 14564\} \qquad (2-14)$$

For $QP$ = 0 to 5, $QP$/6=0 and hence the corresponding $MF = 26214$. Similarly for $QP/6 = 5$ , corresponding $MF = 14564$.

The Dequantizer equation is also same as the proposal of Samsung Electronics, Ltd, FastVDO (JCTVC-F251). Moreover, Cisco Systems and Texas Instrument Inc. proposed the same quantization and dequantization equation in their previous proposals in JCTVC-F446 [40] and JCTVC-E243 [41] which are similar to the proposal of Samsung Electronics, LTD, FastVDO [39].

CHAPTER 3

# PROPOSED ALGORITHM

## 3.1 Introduction

This chapter describes the Division free Quantization Algorithm (DFQA) which is applicable to five popular video standards (H.264, AVS, VC-1, MPEG-2/4 and MJPEG) and standard HEVC under development. Although each standard defines a specific quantization scheme, here a generalized Division free Quantization Algorithm (DFQA) is developed which is applicable to any of the six standards. This algorithm is completely division free to make it simple in terms of hardware overhead and to be applicable to all standards those define quantization as division operation or multiplication followed by shift operation.

## 3.2 Proposed Division Free Quantization Algorithm (DFQA)

Quantization is the division of the transform coefficient from the transform unit by the corresponding Q-value to reduce the bit rate. But in H.264, AVS and HEVC, it is done by multiplication and right shift operation. Hence these standards define their own Multiplication Factors (*MF*). These MFs are multiplied with the transform coefficients and finally right shifted. However, the quantization in VC-1, MPEG-2/4 and MJPEG is defined as division operation only (using 8x8 matrices). As a result, it is a challenge to establish a relationship that is general enough to merge all these schemes. All the quantization schemes and parameters of six different standards are observed carefully and a relationship is developed among them. To integrate VC-1, MPEG-2/4 and MJPEG with the newer standards such as H.264, AVS and HEVC, the division operation of VC-1, MPEG-2/4 and MJPEG is replaced with multiplication, addition and shift operation.

In light of this observation, an equation to calculate the multiplication factor (*MF*) for VC-1, MPEG-2/4 and MJPEG is developed to make the quantization operation of these standards compatible to those of H.264, AVS and HEVC. Finally a generalized algorithm is developed for all these six standards.

### 3.2.1 Steps of DFQA

Our DFQA is divided into three steps. The third and final stage gives the quantized level. These steps and the parameters are described below using equation (3-1) to (3-3).

Step 1:

$$p = (w[i,j] << r)\,.\,MF[i,j], \text{ for } i,j = 0,\ldots,7; \qquad (3-1)$$

Step 2:

$$q = p + (offset << qbits); \qquad (3-2)$$

Step 3:

$$y = q >> (n + qs\_bit); \qquad (3-3)$$

Where, $w$ denotes the transform coefficient, $MF$ is the Multiplying factor and $y$ denotes the corresponding quantized value (level), $<< r = \text{left shift by } r \text{ bits}$, $>> (n + qs\_bit) = \text{right shift by } (n + qs\_bit) \text{ bits}$. $QP$ is the Quantization Parameter which specifies $MF$s, $M = \log 2(N)$, $N$ is the transform size, $DB = B - 8$, $B$ is source bit width (8 or 10 bits according to [33]). In the next sections, the general DFQA is applied to individual codecs including examples for each of them.

### 3.2.2 Equation to calculate *MF* for VC-1, MPEG-2/4 and MJPEG

The proposed algorithm exploits similar quantization parameters and equations for VC-1 and MPEG-2 [9]. Moreover, according to [12] MPEG-4 part 2 can also use similar quantization approach as it supports custom quantization table that can be tuned to content and bit rate like MPEG-2. So, in the proposed algorithm the same quantization scheme for both MPEG-2 and MPEG-4 part 2 is used and referred to as the quantization of MPEG-2/4. As a result, the value of *MF* of VC-1 and MPEG-2/4 is similar in this work. However, the *MF* is different for MJPEG standard. The *MF* for VC-1/MPEG-2/4 is calculated from the intra matrix described in Chapter 2 by equation (2-6). The *MF* for MJPEG is calculated from the luminance matrix shown by equation (2-8). For the calculation of *MF*, we follow the simple equation given by (3-4):

$$MF(i,j) = \text{round}((2 \wedge n) / qm(i,j)); \text{where } i,j = 0,1,\ldots,7 \qquad (3-4)$$

Where $n$ is the same parameter used in (3-3) which are described in Table 3.1 and $qm(i,j)$ is the Q matrix coefficient described in equation (2-6) and (2-8) for VC-1/MPEG-2/4 and MJPEG respectively. The example of *MF* calculations for VC-1/MPEG-2/4 and MJPEG are given in section 3.2.5 and 3.2.6 respectively.

### 3.2.3 DFQA applied to H.264

In this section, the generalized DFQA is applied to perform the quantization operation in H.264. Firstly, the transform coefficients coming from the transform unit are directly multiplied by $MF$ as the value of $r$ is equal to 0 for this standard and hence no left shift operation is applied to the transform coefficients. In the second stage, $offset$ is left shifted by $qbits$ and then added to the result coming from the first stage. Here, the $offset$ value ranges between 0 and 0.5 as specified by the standard. Moreover $qbits$ is specified as $(16 + QP/6)$. Here, $QP$ is the quantization parameter ranging from 0 to 51 as described in Table 2.1. The value obtained from the second step is finally right shifted by $n + qs\_bit = ((16 + QP \bmod 6) + 0)$ bits in the third stage which is the final stage of the proposed DFQA. Hence for this standard, $n = (16 + QP / 6)$ and $qs\_bit = 0$. In case of H.264 as specified in [7], Multiplication Factor, $MF$ depends on m (= QP/6) and the position $(i, j)$ of the element as described in chapter 2 by Figure 2.2.

### 3.2.3.1 Example

The following example shows how the DFQA (configured for H.264) is applied to the pixel values of Barbara image shown by Figure 3.1. Here, we have used MATLAB for verification. Let us take an arbitrary 8x8 pixel matrix from Barbara image referred to as $image\_in$.

Figure 3.1. Barbara image

$$image\_in = \begin{bmatrix} 143 & 139 & 134 & 127 & 122 & 114 & 105 & 95 \\ 90 & 85 & 78 & 76 & 69 & 71 & 67 & 70 \\ 76 & 74 & 83 & 91 & 97 & 104 & 112 & 120 \\ 107 & 126 & 134 & 144 & 151 & 152 & 161 & 166 \\ 143 & 171 & 177 & 184 & 189 & 188 & 189 & 197 \\ 156 & 194 & 197 & 203 & 204 & 203 & 205 & 200 \\ 147 & 196 & 198 & 201 & 198 & 195 & 192 & 184 \\ 117 & 168 & 168 & 165 & 159 & 151 & 148 & 136 \end{bmatrix} \qquad (3-5)$$

The next step is to apply a 1-D DCT operation on this $image\_in$ matrix according to [15]. The transformed coefficients are shown by the matrix called $H.264\_trans$ .

$$H.264\_trans = \begin{bmatrix} 6921 & 7804 & 7906 & 8085 & 8095 & 8071 & 8104 & 8140 \\ -2876 & -4392 & -4590 & -4798 & -4897 & -4824 & -4921 & -4822 \\ 1766 & 2276 & 2039 & 1775 & 1464 & 1273 & 981 & 564 \\ 876 & 527 & 416 & 370 & 353 & 323 & 197 & 222 \\ 1308 & 1681 & 1648 & 1542 & 1487 & 1285 & 1197 & 1058 \\ -22 & -244 & -223 & -199 & -144 & -169 & -149 & -212 \\ 1123 & 1336 & 1398 & 1398 & 1433 & 1352 & 1401 & 1206 \\ 47 & -92 & -18 & -45 & 17 & -15 & -72 & -76 \end{bmatrix} \qquad (3-6)$$

For $QP = 0$ each coefficient of $H.264\_trans$ is multiplied by the corresponding value of $MF\_0$ described in chapter 2, equation (2-2). This multiplication is a point-wise multiplication (instead of a matrix multiplication) and is added to the left shifted $offset$ value. For $QP = 0$ and $offset = 0$, the final quantized output after the right shift operation is given below:

$$H.264\_quant = \begin{bmatrix} 1384 & 1455 & 2024 & 1508 & 1619 & 1505 & 2075 & 1518 \\ -535 & -766 & -1084 & -837 & -913 & -841 & -1162 & -840 \\ 452 & 538 & 652 & 419 & 375 & 301 & 314 & 133 \\ 163 & 92 & 98 & 65 & 66 & 56 & 47 & 39 \\ 261 & 313 & 422 & 288 & 297 & 240 & 302 & 197 \\ -4 & -43 & -53 & -35 & -27 & -29 & -35 & -37 \\ 287 & 316 & 447 & 330 & 367 & 319 & 448 & 285 \\ 9 & 16 & -4 & -8 & 3 & -3 & -17 & -13 \end{bmatrix} \qquad (3-7)$$

### 3.2.4 DFQA applied to AVS

Next, the DFQA is applied to perform the quantization operation of AVS. For AVS, based on [28], $MF$ depends on $QP$ where the range for $QP$ is 0 to 63. Each $QP$ specifies one particular $MF$. The value of $MF$ for the particular $QP$ is given by Table 2.3 in Chapter 2. Once $QP$ is specified the corresponding $MF$ is multiplied with the transform coefficient in the first step of the DFQA. Again in case of AVS the transform coefficient is not left shifted by $r$ bits as the value of $r$ is 0 which is similar to the case of H.264. After that in second step this result is added to the 14 bits left shifted $offset$ value. The $offset$ value is 1 in case of AVS standard. Finally in the last step of DFQA the value of step 2 is right shifted by $n + qs\_bit$, where $n = 15$ and $qs\_bits = 0$. In the end, in the third step, a 15-bit right shift operation is performed to get the final quantized output.

#### 3.2.4.1 Example

The following example shows how the DFQA (configured for AVS) is applied to Barbara image to get the quantized value in AVS using the AVS quantization parameters. Here the same 8x8 image matrix from Barbara referred to as $image\_in$ is used. Now this $image\_in$ matrix is 1-D transformed according to get the transformed coefficient based on the operation specified in [15]. The transform coefficients are given by the following matrix named as $AVS\_trans$.

$$AVS\_trans = \begin{bmatrix} 6669 & 7488 & 7614 & 7806 & 7841 & 7834 & 7885 & 7938 \\ -2832 & -4355 & -4536 & -4726 & -4803 & -4727 & -4801 & -4694 \\ 2001 & 2558 & 2338 & 2097 & 1799 & 1613 & 1334 & 928 \\ 712 & 436 & 311 & 239 & 204 & 143 & -3 & -16 \\ 1168 & 1476 & 1421 & 1309 & 1248 & 1049 & 931 & 816 \\ 331 & 258 & 270 & 283 & 313 & 271 & 266 & 160 \\ 500 & 515 & 552 & 522 & 547 & 475 & 513 & 342 \\ 603 & 520 & 606 & 598 & 659 & 638 & 595 & 606 \end{bmatrix} \qquad (3-8)$$

Once the transform coefficients are obtained, the DFQA is applied to the transform coefficient matrix, in this case $AVS\_trans$. To apply DFQA, at first the Quantization Parameter $QP$ is chosen. Let us choose $QP = 0$, and then the corresponding $MF$ is 32768. The transform coefficient matrix is multiplied by $MF = 32768$ if $QP = 0$, is chosen or by $MF = 29775$ $QP = 1$ is chosen, and so on. For this experiment, $QP = 10$ is used. As a result, the corresponding $MF$ is 13777 which is multiplied by the each of the transform coefficient of $AVS\_trans$ matrix. First few values are calculated as: $6669 * 13777 = 91878813$, $7488 * 13777 = 103162176$ and so on. Then, in second step, these values are added with the left shifted *offset* value. The values obtained in the second step are: $91878813 + (1 << 14) = 91895197$, $103162176 + (1 << 14) = 103162204$ and so on. Finally these values are right shifted by 15 bits in step three. The final values are 2804, 3148 and so on. The whole quantized matrix of AVS standard by applying DFQA is shown by $AVS\_quant$ matrix.

$$AVS\_quant = \begin{bmatrix} 2804 & 3148 & 3201 & 3282 & 3297 & 3294 & 3315 & 3338 \\ 1190 & 1830 & -1906 & -1986 & -2018 & -1986 & -2018 & -1973 \\ 841 & 1076 & 983 & 882 & 756 & 678 & 561 & 390 \\ 299 & 183 & 131 & 100 & 86 & 60 & -0.761 & -6.22 \\ 491 & 621 & 597 & 550 & 525 & 441 & 391 & 343 \\ 139 & 108 & 114 & 119 & 132 & 114 & 112 & 67 \\ 210 & 217 & 232 & 219 & 230 & 200 & 216 & 144 \\ 254 & 219 & 255 & 251 & 277 & 268 & 250 & 255 \end{bmatrix} \quad (3-9)$$

### 3.2.5 DFQA applied to VC-1 and MPEG-2/4

VC-1 uses multiple transform sizes but the same quantization rule is applied to all the coefficients. At a high level the quantization process (scalar quantization where each transform coefficient is independently quantized and coded) in VC-1 is similar to the corresponding process in MPEG-2 standard. According to [12], the similar quantization scheme can be for MPEG-2 and

MPEG-4 part-10. As specified by [7], the MPEG-2 standard uses two quantization matrices, Intra matrix and non-intra matrix. However for simplicity only the Intra matrix is considered. The non intra quantization can be implemented just by adding an extra look-up Table.

However, $MF$ for each of the coefficient of the quantization matrix shown by equation (2-6) is calculated based on equation (3-4). This $MF$ is then right shifted by 8 bits which is an approximation to the division. For example, dividing a DCT coefficient ($w$) coming from transform unit by a quantization value 19 (quantization matrix value coming from Figure 2-4) can be expressed as:

$$\frac{w}{19} = \frac{w}{19} * \frac{14}{14} \approx \frac{w*14}{2^8} \qquad (3-10)$$

Similarly, $MF$ for quantization value 22 can be derived by (3-11):

$$\frac{w}{22} = \frac{w}{22} * \frac{12}{12} \approx \frac{w*12}{2^8} \qquad (3-11)$$

Hence for 19, the corresponding $MF$ is 14 and for 22 the corresponding $MF$ is 12. The same strategy is followed to calculate all the $MF$. Moreover according to [7] the quantization in VC-1 and MPEG/2-4 needs the denominator of (3-10) and (3-11) to be multiplied by quantization step ($QS$) and in this architecture for simplicity $QS = 32 = 2^5$ is chosen, where 5 is denoted as $qs\_bit$. So the right hand side of (3-10) can be characterized as:

$$\frac{w*14}{2^8 * 32} = \frac{w*14}{2^8 * 2^5} = \frac{w*14}{2^{13}} \qquad (3-12)$$

The following matrix shows the $MF$s for the corresponding quantization matrix in equation (2-6):

$$MF = \begin{bmatrix} 31 & 16 & 14 & 12 & 10 & 9 & 9 & 8 \\ 16 & 16 & 12 & 10 & 9 & 9 & 8 & 7 \\ 14 & 12 & 10 & 9 & 9 & 8 & 8 & 7 \\ 12 & 12 & 10 & 9 & 9 & 8 & 7 & 7 \\ 12 & 10 & 9 & 9 & 8 & 7 & 6 & 5 \\ 10 & 9 & 9 & 8 & 7 & 6 & 5 & 4 \\ 10 & 9 & 9 & 8 & 7 & 5 & 4 & 4 \\ 9 & 9 & 7 & 7 & 5 & 4 & 4 & 3 \end{bmatrix} \qquad (3-13)$$

As compared to the elements in the original intra matrix in equation (2-6), the elements in (3-13) are smaller which helps to decrease the size of the RAM. The proposed quantization scheme is also verified in MATLAB. The result shows that the approximation of  including the right shift operation is completely suitable for real time image processing.

Once $MF$ is obtained, in the first step this $MF$ is multiplied with the 4 bits left shifted transform coefficient as the value of  is 4 in this case. However, second step is not applicable to this standard as this standard does not need any addition operation. Hence both *offset* value and *qbits* are 0 in this case. The output of the first step is directly right shifted by $(n + qs\_bit) = (8+5) = 13$ bits in the third step.

### 3.2.5.1 Example

The following example shows how the proposed DFQA is applied to VC-1 and MPEG-2/4 standard where the input image matrix is $image\_in$. At first like the other standards 1-D forward transform is performed. However VC-1 and MPEG-2/4 has different transform matrix [15]. So the transform coefficients are different for VC-1 and MPEG-2/4. However the architecture of DFQA defines similar quantization for them. First four transform coefficients for VC-1 and MPEG-2/4 are 10447, 11748, 11951, 12257 and 315058, 353880, 359637, 368669 respectively. Once the transform coefficients are obtained the first step of DFQA is applied to the transform coefficient. For VC-1, the first two quantized values are $((10447 << 4)*31+0) >> 13 = 634$ and $((11748 << 4)*16+0) >> 13 = 367$ respectively where 10447 and 11748 are the transform coefficients which are left shifted by 4 bits, 31 and 16 are the $MF$ coming from the matrix of (3-13). For MPEG-2/4 the same $MF$ matrix of (3-13) is used only the transform coefficients are different and the quantized outputs can be calculated as VC-1.

### 3.2.6 DFQA applied to MJPEG

Similarly for Motion JPEG codec, $MF$s for each of the coefficients of the Luminance and Chrominance quantization matrix are calculated. Also, the work in [8] uses the same quantization matrix of Motion JPEG to implement a pixel based post-processing algorithm to enhance the quality of Motion JPEG. However, the image quality can be varied by varying the quality factor. The lower the quality factors the more the compression. The work in [38] shows how the varying quality factor changes the JPEG image quality. This design is only focused on the Luminance matrix. The $MF$ for luminance and chrominance quantization matrix is calculated in the exact same way as the VC-1 and MPEG-2/4 described in the previous section. Equation (3-14) and (3-15) shows the $MF$ matrices for luminance and chrominance quantization matrices respectively.

$$MF\_LUMINANCE = \begin{bmatrix} 16 & 22 & 25 & 16 & 10 & 6 & 5 & 4 \\ 20 & 20 & 18 & 14 & 10 & 4 & 4 & 5 \\ 18 & 20 & 16 & 10 & 6 & 4 & 4 & 4 \\ 18 & 15 & 11 & 8 & 5 & 3 & 3 & 4 \\ 14 & 11 & 7 & 4 & 4 & 2 & 2 & 3 \\ 10 & 7 & 5 & 4 & 3 & 2 & 2 & 3 \\ 5 & 4 & 3 & 3 & 2 & 2 & 2 & 2 \\ 4 & 3 & 3 & 3 & 2 & 2 & 2 & 2 \end{bmatrix} \qquad (3-14)$$

$$MF\_CHROMINANCE = \begin{bmatrix} 15 & 14 & 10 & 5 & 2 & 2 & 2 & 2 \\ 14 & 12 & 10 & 4 & 2 & 2 & 2 & 2 \\ 10 & 10 & 4 & 2 & 2 & 2 & 2 & 2 \\ 5 & 4 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \end{bmatrix} \qquad (3-15)$$

Again the elements in the $MF$ matrices are smaller than those of the original Luminance and chrominance matrix which reduces the size of the RAM. After that, each $MF$ is directly multiplied by the corresponding transform coefficient in the first step as $r = 0$. Similar to VC-1 and MPEG-2/4 codecs, as the *offset* and *qbits* are 0, the second step is not applied to the MJPEG standard. Finally in third step of the DFQA the multiplied output of the initial step is right shifted by $n + qs\_bit$ bits. Here $n = 8$ and $qs\_bit = 0$. Hence in the final step 8 bits right shift is performed on the value coming from the first step which gives the desired quantized level.

### 3.2.6.1 Example

The following example shows how the proposed DFQA is applied to MJPEG standard where the input image matrix is the same $image\_in$ matrix. At first like the other standard 1-D forward transform is performed according to the procedure of [15]. First four transform coefficients for MJPEG are 315058, 353880, 359637 and 368669. Once the transform coefficients are obtained the first step of DFQA is applied to the transform coefficient. For MJPEG, the first two quantized values are $((315058 << 0)*16 + 0) >> 8 = 19691$ and

$((353880 << 0)*22+0) >> 8 = 30411$ respectively where 315058 and 353880 are the transform coefficients which are left shifted by 0 bits, 16 and 22 are the $MF$ described in (3-14).

To reduce the quantization error in VC-1, MPEG-2/4 and MJPEG and better approximation, $MF$ is also calculated with different amount of right shift operation. For example the Q matrix value, 19 of equation (2-6) can be approximated by $MF = 27$ with 9 bits right shift operation. In the same way for 26 (Q matrix value of equation (2-6)) the $MF = 39$ while 10 bits right shift operation is required. Hence it is clear that for every Q matrix value the best approximation could be taken with different amount of right shift. This approach needs separate look-up-tables (for both MJPEG and VC-1/MPEG-2/4) to store the value of right shift operation. Moreover, designing the control logic following all these operation needs more hardware. As a result this approach increases the hardware cost and cannot reduce the quantization error significantly. Hence the quantization is implemented by using the $MF$ with fixed 8 bit right shift. Equation (3-16) shows the $MF$ matrix for MJPEG derived from equation (2-8) with different amount of right shift. The amount of right shift is given by equation (3-17). The $MF$ matrix with different amount of right shift can also be calculated for VC-1 and MPEG-2/4 based on equation (2-6). The $MF$ matrix and the amount of right shift for VC-1/MPEG-2/4 are specified by (3-18) and (3-19) respectively.

$$MF\_LUMINANCE\_1 = \begin{bmatrix} 1 & 23 & 13 & 1 & 21 & 13 & 20 & 17 \\ 21 & 21 & 18 & 27 & 20 & 18 & 17 & 19 \\ 18 & 20 & 1 & 21 & 13 & 18 & 15 & 18 \\ 18 & 15 & 23 & 18 & 20 & 47 & 13 & 17 \\ 14 & 23 & 14 & 18 & 15 & 19 & 20 & 27 \\ 21 & 15 & 19 & 1 & 13 & 20 & 18 & 11 \\ 21 & 1 & 13 & 47 & 20 & 17 & 17 & 20 \\ 14 & 11 & 11 & 21 & 18 & 21 & 20 & 21 \end{bmatrix} \qquad (3-16)$$

$$SR\_LUMINANCE = \begin{bmatrix} 4 & 8 & 7 & 4 & 9 & 9 & 10 & 10 \\ 8 & 8 & 8 & 9 & 9 & 10 & 10 & 10 \\ 8 & 8 & 4 & 9 & 9 & 10 & 10 & 10 \\ 8 & 8 & 9 & 9 & 10 & 12 & 10 & 10 \\ 8 & 9 & 9 & 10 & 10 & 11 & 11 & 11 \\ 9 & 9 & 10 & 6 & 10 & 11 & 11 & 10 \\ 10 & 6 & 10 & 12 & 11 & 11 & 11 & 11 \\ 10 & 10 & 10 & 11 & 11 & 11 & 11 & 11 \end{bmatrix} \qquad (3-17)$$

$$MF\_INTRA = \begin{bmatrix} 1 & 1 & 27 & 23 & 20 & 19 & 18 & 15 \\ 1 & 1 & 23 & 21 & 19 & 18 & 15 & 14 \\ 27 & 23 & 20 & 19 & 18 & 15 & 15 & 14 \\ 23 & 23 & 20 & 19 & 18 & 15 & 14 & 13 \\ 23 & 20 & 19 & 18 & 1 & 15 & 13 & 21 \\ 20 & 19 & 18 & 1 & 15 & 13 & 21 & 18 \\ 20 & 19 & 18 & 15 & 14 & 11 & 18 & 15 \\ 19 & 18 & 15 & 14 & 11 & 18 & 15 & 25 \end{bmatrix} \qquad (3-18)$$

$$SR\_INTRA = \begin{bmatrix} 3 & 4 & 9 & 9 & 9 & 9 & 9 & 9 \\ 4 & 4 & 9 & 9 & 9 & 9 & 9 & 9 \\ 9 & 9 & 9 & 9 & 9 & 9 & 9 & 9 \\ 9 & 9 & 9 & 9 & 9 & 9 & 9 & 9 \\ 9 & 9 & 9 & 9 & 5 & 9 & 9 & 10 \\ 9 & 9 & 9 & 5 & 9 & 9 & 10 & 10 \\ 9 & 9 & 9 & 9 & 9 & 9 & 10 & 10 \\ 9 & 9 & 9 & 9 & 9 & 10 & 10 & 11 \end{bmatrix} \qquad (3-19)$$

### 3.2.7 DFQA applied to HEVC

As HEVC is a developing new standard and not yet been finalized, the proposals of the quantization algorithm are observed carefully to find out the consistency. The proposal of Samsung Electronics, Ltd, FastVDO (JCTVC-F251) and the proposal of Cisco Systems and Texas Instrument Inc. (JCTVC-G495, JCTVC-F446, JCTVC-E243) are similar as described in Chapter 2. Hence the algorithm proposed by Cisco Systems and Texas Instrument Inc. (JCTVC-

G495) is chosen. HEVC is an emerging standard and uses several transform sizes ranging from 4x4 to 32x32. However the quantization/dequantization scheme is the same for all transform sizes where the multiplier depends on the value of $QP$ and the shifts depend only on $M = \log 2(N)$. The value of $QP$ can vary from 0 to 51. This architecture applies DFQA to the HEVC standard in the same way as the other standards. However the proposed design can be changed if any further modifications take place in this developing standard. The value of $MF$ depends on $(QP/6)$. Once $MF$ is specified it is multiplied with the transform coefficient in the initial stage of DFQA. The transform coefficient is not left shifted as $r = o$ even in the case HEVC. In the following stage this product is added to the $qbits$ left shifted $offset$ value. For HEVC, the value of $offset = 1$ and $qbits = M - 2 + DB$. For 8x8 transform $M = \log 2(N) = 3; DB = B - 8; B = \text{source bitwidth}$ (8 or 10 bits according to [33]). For this architecture supporting 8x8 quantization, $qbits = 3 - 2 + 2 = 3$ for $B = 10$. Finally this value is right shifted by $(n + qs\_bit) = (21 + (QP \bmod 6 - M - DB))$; where $n = 21$ bits. According to [33], the value of $MF$ for the HEVC standard is specified as follows:

$$MF = \{26214, 23302, 20560, 18396, 16384, 14564\} \qquad (3-20)$$

For $QP = 0$ to 5, $QP/6 = 0$ and the corresponding $MF$ is 26214. Similarly for $QP/6 = 5$ the corresponding $MF$ is 14564. When $(QP/6) > 5$ the factor $MF$ remains unchanged only the value of $qs\_bits$ will change. For example, $qs\_bit = (0 - M - DB)$, for $0 \le (QP/6) \le 5$ and so on.

### 3.2.7.1 Example

Let us continue with the same example of Barbara image, $image\_in$. At first, a 1-D DCT for HEVC [34] is applied to the image matrix to get the transform coefficient. Few values of the transform coefficients are 55854, 62777, 63787, and so on. In the first step, each transform coefficient is multiplied by the $MF$ based on $QP/6$. For example, the quantized value of the first and second transform coefficient can be obtained by the application of DFQA with $QP/6=2$ and $MF=20560$. The quantized value for the first transform coefficient is as follows:

$$\left(55854*20560\right)+(1\ll 3))\gg(21+0-3-2)=\left(55854*20560\right)+(1\ll 3))\gg 16=17522$$

The quantized value for the second transform coefficient is as follows:

$$\left(62777*20560\right)+(1\ll 3))\gg(21+0-3-2)=\left(62777*20560\right)+(1\ll 3))\gg 16=19694$$

Similarly, all the quantized value for each transform coefficient can be found using the proposed DFQA. From the above discussion the summary of all the parameters used in the DFQA are presented in Table 3.1.

Table 3.1. Description of different parameters used in the DFQA

| Parameter | AVS | H.264 | HEVC | VC-1 | MPEG-2/4 | MJPEG |
|---|---|---|---|---|---|---|
| *r* | 0 | 0 | 0 | 4 | 4 | 0 |
| *offset* | 1 | 0 - 0.5 | 1 | 0 | 0 | 0 |
| *qbits* | 14 | 16+ (*QP*/6) | *M*-2+*DB* | 0 | 0 | 0 |
| *n* | 15 | 16+ (*QP*/6) | 21 | 8 | 8 | 8 |
| *qs_bit* | 0 | 0 | (*QP*/6)-*M*-*DB* | 5 | 5 | 0 |

# CHAPTER 4

# HARDWARE IMPLEMENTATION

## 4.1 Introduction

This chapter describes the hardware implementation and overall operation of the proposed multi quantizer scheme. Figure 4.1 shows the traditional quantization using division operation whereas Figure 4.2 shows the quantization of our multi quantizer using the multiplication, addition and shift operation.

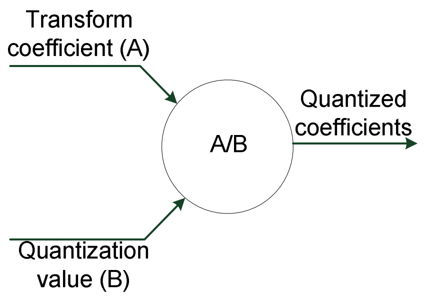

Figure 4.1. Basic quantization scheme using division operation

According to the proposed DFQA presented in Figure 4.2, transform coefficient, multiplication factor, offset and qbits depend on the six different standards based on select_standard pin. The next section describes the detailed hardware architecture of the multi quantizer unit.

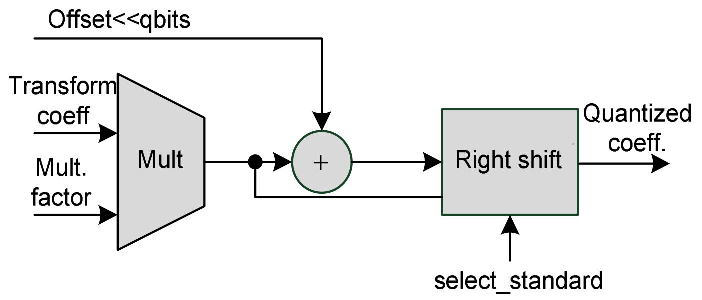

Figure 4.2. Quantization scheme of the proposed multi quantizer

## 4.2 Overall hardware architecture

The multi quantizer unit of the DFQA accepts the transform coefficient, $w$ from the transform unit, multiplication factor $MF$ form the Look-up Table (LUT), $QP$, $offset$, $n$ and $qbits$ from the QP processing unit. The transform unit has been implemented in one of our previous works [15]. Hence the multi quantizer unit accepts the transform coefficients from the multi transform unit presented in [15]. However $QP$ processing is not necessary for hardware implementation as it does not process data but calculates parameters used for data processing in the quantization block. Hence it is assumed that $QP$ processing is previously done by software. The whole architecture is controlled by one Finite State Machine (FSM). The FSM or the control logic generates the desired control signals to control the whole architecture. The overall architecture of our cost-sharing algorithm is shown by Figure 4.3.

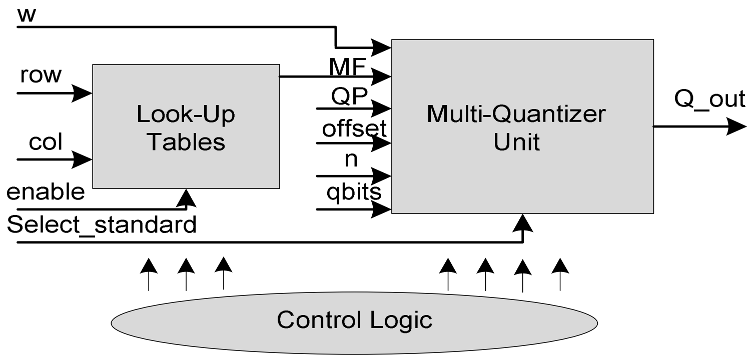

Figure 4.3. Overall block diagram of the DFQA

Moreover Figure 4.4 shows the orientation of the multi quantizer unit, multi transform unit, QP processing unit and the Look-up Tables (LUT).

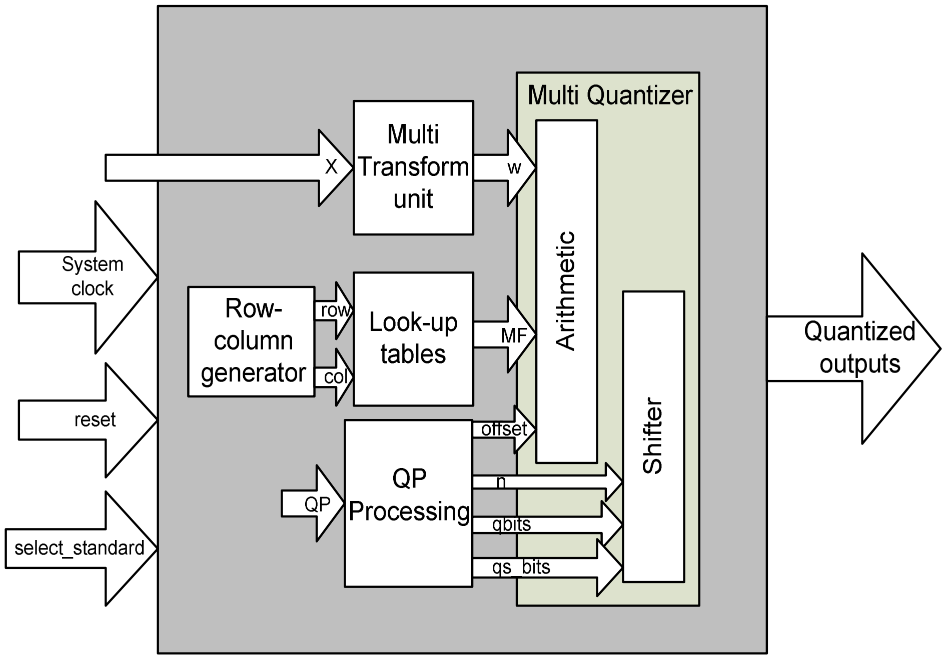

Figure 4.4. Orientation of multi quantizer, multi transform, QP processing and LUT

The proposed architecture can perform the 8x8 quantization of any of the six different standards as selected by the user (or another master system) using the select_standard pin. This select_standard pin has three bits to choose among the six standards. This architecture contains three main blocks with four stage pipelining: (1) look-up tables to hold the multiplying factors ($MF$), (2) one Multi-quantization unit (composed of only one shared multiplier), and (3) one finite state machine to control all the six standards. The description of these blocks and their operations are described in the subsequent sections.

## 4.2.1 Architecture of the multi quantizer

This section describes how the hardware is designed to apply the generalized DFQA to perform the quantization operation in all the six standards. As described in Chapter 3, the DFQA is divided into three steps to get the quantized coefficients. In the first step, the transform coefficients $w$ coming from the transform unit are left shifted by $r$ bits depending on the standard as described in Table 3.1. Then the left shifted transform coefficients are multiplied by the multiplying factors $MF$ from the loop-up tables. Again $MF$ depends on a specific standard as described in Chapter 3. A multiplexer is used to choose the desired $MF$ for a particular standard. The select_standard pin which has three bits chooses the quantization parameters such as $r$, $MF$ etc. among the six standards. The hardware scenario of step 1 of DFQA is illustrated by Figure 4.5.

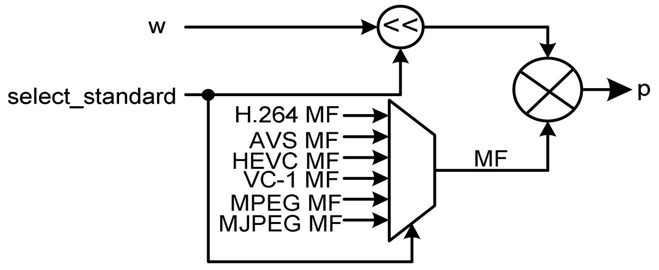

Figure 4.5. Hardware of step 1 of the DFQA

Next in step 2, the $offset$ value is left shifted by $qbits$ and added to the value coming from the first step. In the final and third step the output of step 2 is right shifted by $(n + qs\_bit)$ bits to generate the quantized output. These two steps of DFQA are illustrated by Figure 4.6 and Figure 4.7. Hence from Figure 4.5, Figure 4.6 and Figure 4.7 it is revealed that the core unit of the multi quantizer consists of one general purpose multiplier, shared adder, left shifter and right shifter where the shift operation depends on the select_standard pin.

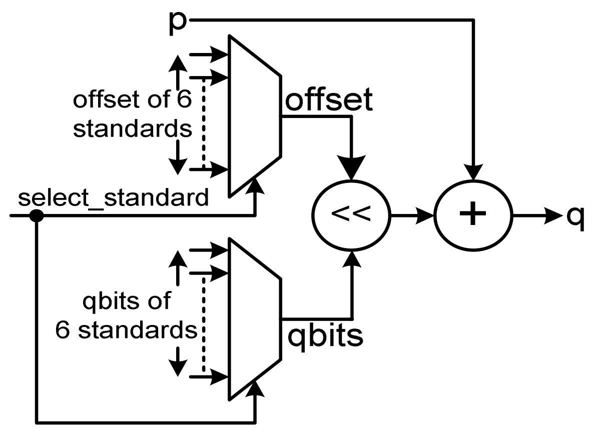

Figure 4.6. Hardware of step 2 of the DFQA

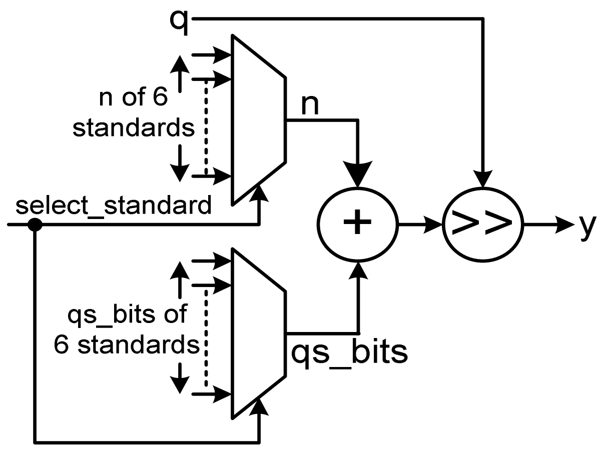

Figure 4.7. Hardware of step 3 of the DFQA

### 4.2.2 Architecture of the look-up tables

The look-up tables in Figure 4.8 contain all the *MF* matrices for six standards. LUT_H.264, LUT_AVS, LUT_VC-1/MPEG-2/4, LUT_MJPEG and LUT_HEVC contains *MF* for H.264, AVS, VC-1/MPEG-2/4, MJPEG and HEVC standard respectively. However one look-up table is used at a time based on the specified standard. The enable pin enables the desired look-up table. The multiplexer is used to select the valid data from the look-up tables. To reduce the power consumption only one look-up table is activated at a time by the enable pin. Once the standard is chosen by the user the desired look-up table is activated by the controller and all the other look-up tables go into the sleep mode. The control logic assigns the enable signal as well as the MUX selection signals accordingly.

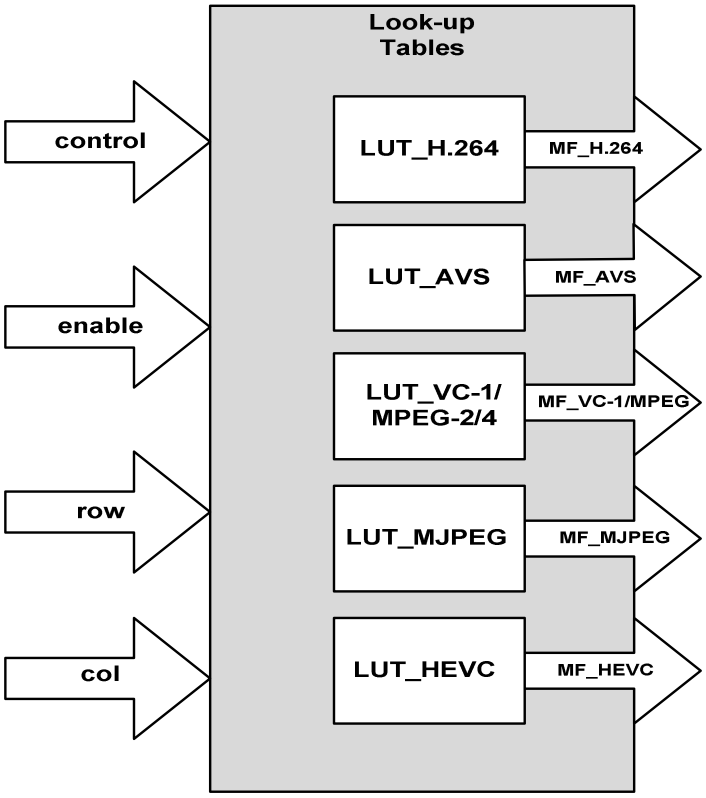

Figure 4.8. Look-up tables of all six standards

For H.264 standard according to Figure 4.9, there are six look-up tables – LUT_H_0, LUT_H_1, LUT_H_2, LUT_H_3, LUT_H_4 and LUT_H_5 for $QP = 0,1,....,5$. If $QP$ is greater than 5 even then the same look-up tables will be used, only the $qbits$ will be changed. For example, with $QP = 6$ and $QP = 7$ LUT_H_0 and LUT_H_1 will be used respectively. Similarly the rest of the look-up tables will be reused as the value of $QP$ is being increased. One multiplexer is used to select the desired LUT among the six LUTs of H.264.

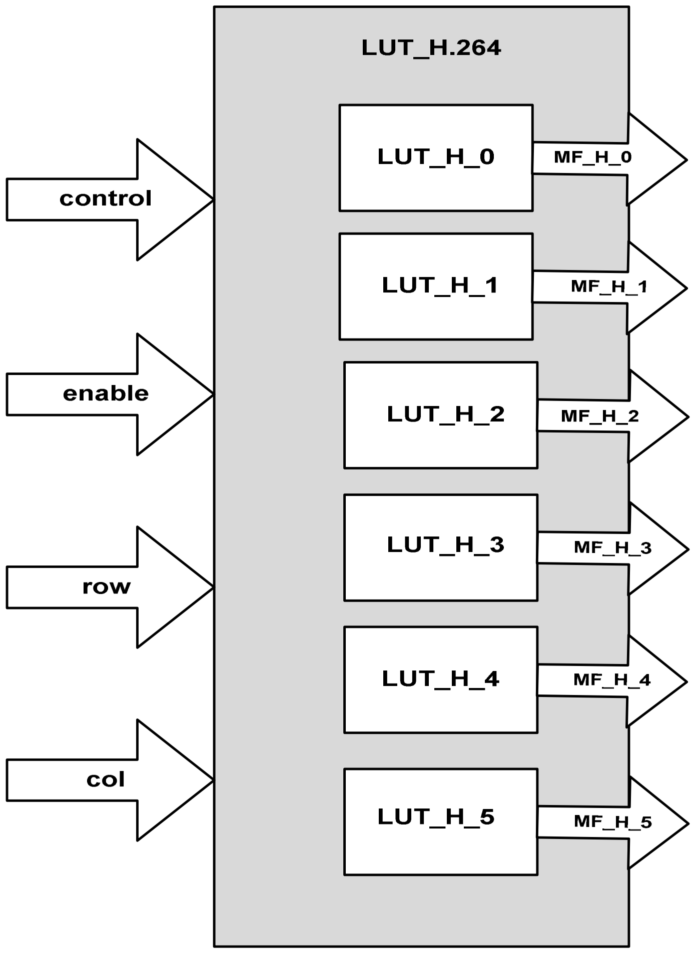

Figure 4.9. Look-up tables of H.264 standard

The entire orientation of these look-up tables and the data flow is demonstrated by Figure 4.10 where MUX1 selects the desired $MF$ for H.264 standard from these six look-up tables based on $QP$. Finally MUX2 is used to select $MF$ of H.264 or HEVC or AVS or VC-1 or MPEG-2/4 or MJPEG. The row and col signal used in Figure 4.8, Figure 4.9 and Figure 4.10 are coming from the row-column-generator to point to the row and column respectively of the Look-up tables.

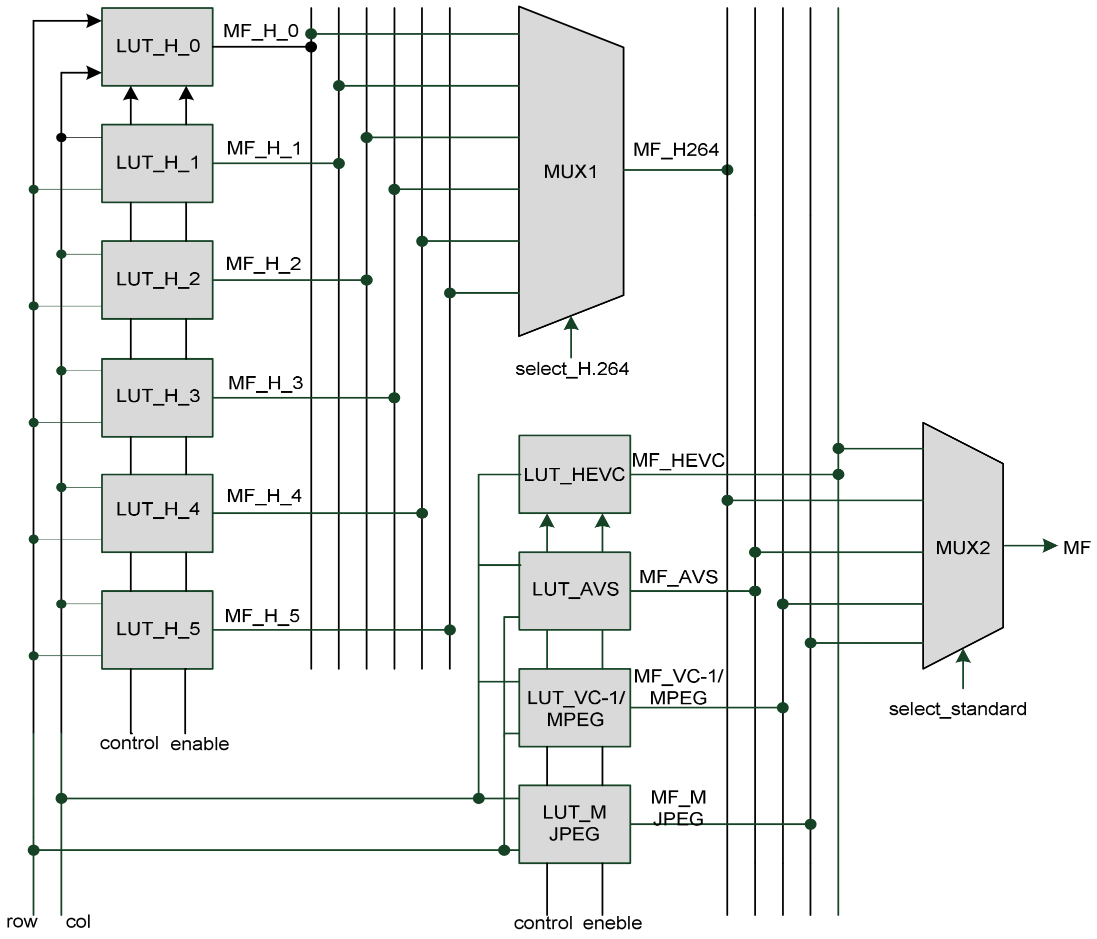

Figure 4.10. Look-up tables and complete data flow diagram

## 4.3 Overall operation

The detailed block diagram of the entire operation with all the pipelining boundaries is shown in Figure 4.11. According to this diagram the our architecture operates in four stages of pipeline which consists of a row-column-generator to point the row and column of the look-up tables, several multiplexers to select the valid path, a shared multiplier for all six standards, a shared adder and shared shifter. Here quantization is performed by multiplication and right shift operation rather than the division operation.

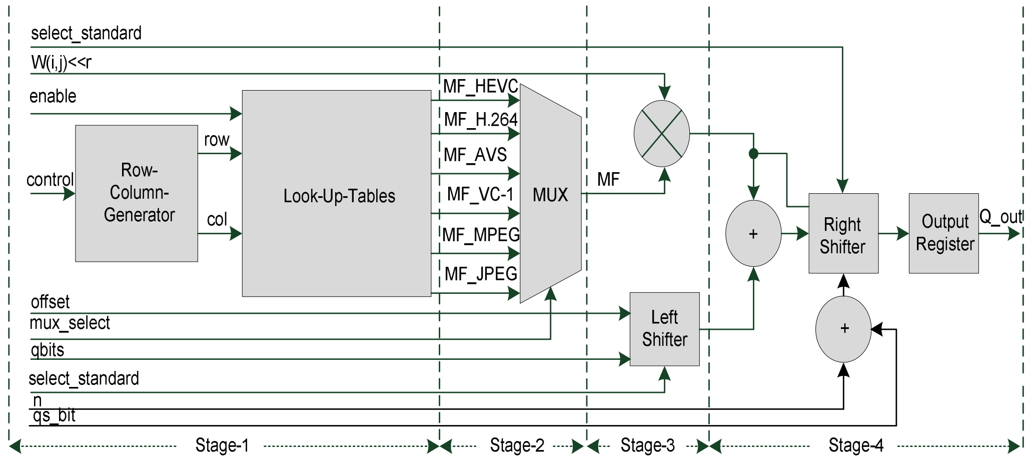

Figure 4.11. Entire operation with all pipelining boundaries

In the first stage of pipelining the value of row and col is generated to point to the row and column of the look-up table by the Row-Column-Generator with the help of the controller. This row and col is used to get the Multiplication Factor, $MF$ from look-up tables. After that in second stage of the pipeline operation the multiplexer selects the valid $MF$. Next in stage 3 the multiplier multiplies the transform coefficient $w[i, j]$ coming from the transform unit with the desired $MF[i, j]$ from the LUTs. Moreover at the same time in this stage the $offset$ value is left shifted by $qbits$ by the left shifter only for H.264, AVS and HEVC standard. For VC-1, MPEG-2/4 and MJPEG this left shifter involved in stage 3 is not used. The output of the multiplier and the left shifter are added and finally right shifted in the fourth and final stage of the pipeline operation and the output register gives the quantized output. However for VC-1, MPEG-2/4 and MJPEG the output of the multiplier is directly right shifted in this final stage and the output register returns the final quantized level. The left shift and right shift operation for AVS, H.264, HEVC, VC-1, MPEG-2/4 or MJPEG is chosen by the select_standard pin. In addition to the logic shown in Figure 4.11 there are pipelining registers between each stage of pipelining.

Initially the entire operation inside the multi quantizer takes five clock cycles to complete the initialization process. First quantized output (Q0) is available at fifth clock cycle (cc) as shown by Figure 4.12. As the architecture operates in four stages of pipelining there are some latencies involved in the operation. The next quantized output, Q1 takes another five clock cycles after Q0 and so on.

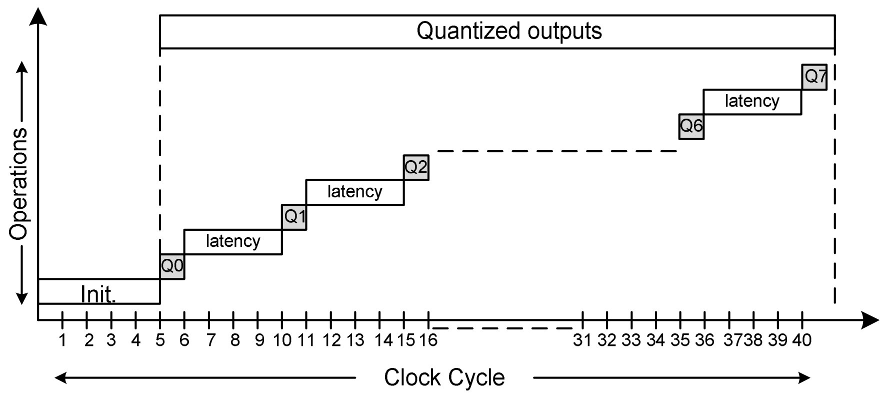

Figure 4.12. Timing diagram of the multi quantizer

The design of the shared multiplier along with the shifter instead of the divider is the key part of the entire process. To integrate the old standards like MPEG-2/4 and MJPEG with the newer standards such as AVS, HEVC and H.264 the whole architecture is proposed as a shared multiplication and right shift operation. Due to this strategy it needs only a shared multiplier rather than both the multiplier (for AVS, HEVC and H.264) and the divider (for VC-1, MPEG-2/4 and MJPEG) which reduces the hardware complexity as well as the cost. Moreover the design shares only one control circuit rather than using specific control circuit for each standard which makes the design more cost effective.

CHAPTER 5

# HARDWARE COMPARISON

## 5.1 Introduction

In this chapter the resource utilization summary is presented. Moreover the performance of the multi quantizer is compared with the existing design in terms of hardware count, operating frequency and decoding capability. This architecture is implemented in Verilog HDL and the operation is verified using Xilinx Virtex4 LX60 FPGA. The design is later synthesized using 0.18 μm CMOS technology. The following sections present the performance comparisons in terms of FPGA as well as VLSI.

## 5.2 Performance comparison in FPGA

The architecture of multi quantizer is implemented in Xilinx Virtex4 LX60 FPGA. It consumes 708 4-input LUTs, 381 slices and 185 registers with a maximum operating frequency of 187.3 MHz. The design is also synthesized in XilinxVirtex2 and Virtex5 FPGA. For better evaluation, the multi-transform design (for five codecs) in [15] is integrated with the multi-quantization scheme. The subsequent sections demonstrate the resource utilization summary as well as the performance comparison of the proposed hardware.

### 5.2.1 Resource utilization summary in FPGA

In Table 5.1, Table 5.2 and Table 5.3 the resource utilization summary of the multi quantizer as well as the combined multi transform and multi quantizer is presented while implemented in Virtex2, Virtex4 and Virtex5 respectively.

Table 5.1. Resource utilization in Virtex2 FPGA

| Arch. | Virtex2 | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | # of register | # of slice | # of LUT | # of latches | # of I/Os | # of IOBs | # of DSP48s | Freq. (MHz) |
| 6 Q | 228 | 394 | 735 | 264 | 250 | 249 | 2 | 142 |
| 5 T + 5 Q | 1079 | 1028 | 1734 | 1079 | 58 | 57 | 2 | 136 |

Table 5.2. Resource utilization in Virtex4 FPGA

| Arch. | Virtex4 | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | # of register | # of slice | # of LUT | # of latches | # of I/Os | # of IOBs | # of DSP4s | Freq. (MHz) |
| 6 Q | 185 | 381 | 708 | 221 | 250 | 249 | 2 | 187 |
| 5 T + 5 Q | 1036 | 972 | 1722 | 1036 | 58 | 57 | 2 | 194 |

Table 5.3. Resource utilization in Virtex5 FPGA

| Arch. | Virtex5 | | | | | | |
|---|---|---|---|---|---|---|---|
| | # of register | # of LUT | # of LUT flip-flop pair | # of I/Os | # of IOBs | # of DSP48s | Freq. (MHz) |
| 6 Q | 182 | 516 | 572 | 250 | 249 | 2 | 196 |
| 5 T + 5 Q | 1035 | 1417 | 1629 | 58 | 57 | 2 | 277 |

### 5.2.2 Performance comparison of the multi quantizer

This section compares the performance of the multi quantizer with the existing designs. In Table 5.4 the performance (FPGA only) of the multi quantizer supporting six standards is summarized in terms of hardware count, maximum working frequency, and supporting standards with other designs. The design in [19] presents two sets of architecture of 4x4 transform and quantization of H.264 standard. The first architecture is optimized for area which results less hardware cost. The second set is optimized for speed at the cost of hardware. The area optimized architecture has one quantizer with four stages of pipeline. The quantization operation is performed by using multiplication and right shift operation instead of division operation. The speed optimized architecture has 16 quantizers in parallel to boost up the throughput. Moreover the speed optimized architecture operates in two stages of pipelining. Although replication of structures can increase parallelism and data throughput the overall core complexity increases proportionally. However, only the area optimized architecture is compared with the multi quantizer architecture. As this design can support 4x4 quantization of only one standard, it has less hardware than ours. The architecture presented in [24] is a unified architecture for 4x4 2-D DCT, 2-D Hadamard, basic quantization, 2-D IDCT, 2-D Hadamard and basic inverse quantization. The quantization module operates in three stages of pipeline with distinct matrices dependent on QP values. This architecture contains four look-up tables for QP 0 to 3 as supported by 4x4 quantization of H.264. It has four parallel multipliers each one consulting one look-up table. Moreover it also has four parallel adders. As compared to the design presented in [24] the proposed multi quantizer has only one shared multiplier and one shared adder to support all the six standards. As the quantization unit of [24] has several multipliers and several adders it increases the cost of the architecture. Here the hardware cost of the multi quantizer is compared with the quantization unit of [24]. The authors in [25] present a 4x4 transform and quantization. The quantization unit has several LUTs for containing f, QP/6 and MF. The quantization block has 16 parallel quantizers with 16 multipliers and adders which increase the core computational complexity as well as hardware cost. Even in this case for comparison the quantization unit is considered.

Table 5.4. Comparison with the existing designs (quantizer only)

| Arch. | Mat. size | Tech. | # of LUT | # of slice | Freq. (MHz) | Supporting standard | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | H.264 | AVS | VC-1 | MPEG-2/4 | MJPEG | HEVC |
| [19] | 4x4 | Virtex2 | 286 | 143 | 135 | Y | o | o | o | o | o |
| [24] | 4x4 | Virtex2 | 956 | -- | 108 | Y | o | o | o | o | o |
| [25] | 4x4 | Virtex2 | 1984 | 992 | 94 | Y | o | o | o | o | o |
| Ours | 8x8 | Virtex4 | 708 | 381 | 187 | Y | Y | Y | Y | Y | Y |
| Ours | 8x8 | Virtex2 | 735 | 394 | 142 | Y | Y | Y | Y | Y | Y |

‘Y’- yes; ‘o’- no; ‘--’ – no information;

### 5.2.3 Performance comparison of the transform-quantizer

For better evaluation, the multi transform unit of [15] is integrated with the multi quantizer unit to compare the performance with the existing designs those support both transform and quantization. The combined design is implemented in Xilinx FPGA (Virtex4 LX60). This combined architecture costs 1722 LUTs (4-input), 972 slices and 1036 registers with a maximum operating frequency of 194.7 MHz when synthesized in Virtex4 FPGA. The design is also synthesized in Virtex2 FPGA and all the comparisons are presented in Table 5.5. The performance (FPGA only) of this combined multi DCT and multi quantizer is compared with the existing designs those include both DCT and quantization block in Table 5.5.

The authors of [21] present 8x8 transform and quantization of H.264 standard. The architecture is designed to perform pipelined operation which has mainly three units, (i) transform unit, (ii) QP processing unit and (iii) quantization unit. The transform operation is done using the butterfly network. The QP processing block is responsible for calculating the intermediate variables such as offset value, qbits value, MF values etc for the quantization which needs more hardware. The quantization unit computes the quantized value using multiplication and right shift operation. The architecture in [22] computes 4x4 and 8x8 transform, quantization, inverse transform and inverse quantization of H.264. The transform unit has eight parallel transform subunits each of which operates on one row/column. As it can support both 4x4 and 8x8 operation it consumes more hardware. The quantization unit also operates in parallel processing mode. The quantization module of [22] consists of 32 identical sub-quantizers. To support 32 quantizer sub-units this design needs 32 multipliers. As this design has several multipliers and look up tables for both 4x4 and 8x8 quantization operation which ends up with much higher computation cost. A design to support 4x4 transform, inverse transform, hadamard transform, inverse hadamard transform, quant and inverse quant is presented in [26]. A reconfigurable datapath with one shared multiplier is used in the quantization process. The quantization is done by multiplication, addition and shift operation instead of division operation. The design in [23] presents hardware architecture to support forward transform and quantization of H.264 standard. This design presents unified 1-D transform unit to perform Discrete Cosine Transform and Walsh Hadamard Transform. The quantizer consists of LUTs to contain MF, QP processing unit, multiplier, adders and shifters. The quantization unit has four identical sub-quantizers. The design presented in [27] can support only one standard, H.264 and the implementation is quite straight forward, and hence has less hardware than ours. However none of these designs can support the entire six quantization scheme. Compared to the existing designs, the proposed design has higher frequency of operation with comparable hardware count. Thus, it can be seen from the Table 5.4 and Table 5.5 that this design can support the highest number of popular and widely used video standards (i.e., AVS, H.264/ AVC, VC-1, MJPEG, MPEG-2/4 and the developing new standard HEVC) and still consumes relatively less hardware cost and runs at higher operational frequency.

Table 5.5. Comparison with the existing designs (transform and quantization)

| Arch. | Mat. size | Tech. | # of LUT | # of slice | Freq. (MHz) | Supporting standard | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | H.264 | AVS | VC-1 | MPEG-2/4 | MJPEG |
| [19] | 8x8 | Virtex2 | 2887 | 1624 | 112 | Y | o | o | o | o |
| [21] | 8x8 | Virtex2 | 29018 | 14509 | 69 | Y | o | o | o | o |
| [22] | 8x8 | StratixII | 7323 | -- | 100 | Y | o | o | o | o |
| [23] | 4x4 | Virtex2 | 246 | 123 | 115 | Y | o | o | o | o |
| [26] | 4x4 | Virtex2 | 2498 | 1249 | 81 | Y | o | o | o | o |
| [27] | 4x4 | Virtex2 | 1386 | 693 | 99 | Y | o | o | o | o |
| Ours | 8x8 | Virtex4 | 1722 | 972 | 194 | Y | Y | Y | Y | Y |
| Ours | 8x8 | Virtex2 | 1734 | 1028 | 136 | Y | Y | Y | Y | Y |

'Y'- yes; 'o'- no; '--' – no information;

## 5.3 Performance comparison in VLSI

The quantization and combined transform and quantization unit is synthesized in CMOS 0.18 µm technology using Artisan library cells. 187715 µm2 silicon area, 20.8K gate and 7.2K standard cells are consumed by the multi quantization unit. The frequency of operation is 86.4 MHz. the combined multi transform and quantization unit costs 251,302 µm2 silicon area, 27.9 K logic gates, and 8.2K standard cells. The frequency of operation is 79.7 MHz. As seen here, the operational frequency of our design is much higher in FPGA because of the use of optimized LUTs that are inherent to the type of FPGA chosen.

### 5.3.1 Estimated area savings in the multi quantizer

This section shows an estimated area savings of the multi quantizer as compared to the existing designs in VLSI level. In Table 5.6 the no. of gate count in VLSI level of the multi quantizer is compared with that of the estimated cost of the existing designs. Since, as of today we have not come across to any design that can support all six codecs an estimate of the projected cost is shown.

Table 5.6. Comparison of multi quant with the estimated cost (VLSI only)

| **Codecs** | **# of gates** |
|---|---|
| VC-1[9] | 9 K |
| H.264 [19] | 1.75 K |
| AVS [28] | 7.2 K |
| MPEG-2 [29] | 9 K |
| MJPEG [42] | 4.1 K |
| HEVC (ours) | 9.9 K |
| Estimated total cost (6 codecs - standalone) | 40.95 K |
| Ours (all 6 codecs - shared ) | 20.8 K |

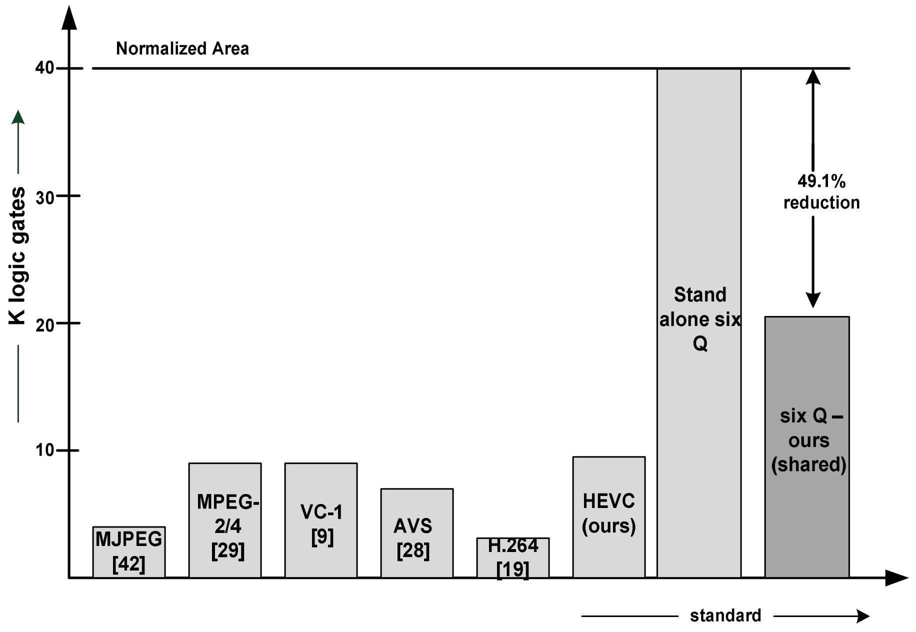

Figure 5.1. Comparison of normalized area of the multi quantizer

Note that the design in [19] has less number of gate counts as it is a 4x4 quantization implementation of H.264. The design in [28] describes 8x8 quantization and inverse quantization of AVS encoder based on zero prediction method. This design predicts whether a transform coefficient can be quantized to zero by a simple formulation. The coefficient predicted to be zero is skipped without quantization. Instead of detecting each transform coefficient this design skips a subset of coefficient quantized to be zero by a rough detection. However, the proposed design is compared with [28] in terms of gate count needed only for the quantizer excluding the zero prediction method. The authors of [29] present VLSI architecture for DPCM Hybrid Coding Loop to support 2-D DCT/IDCT, quantization and inverse quantization of MPEG-2. Again only the hardware cost of the quantization is taken into account to make a fair comparison with our design. The design in [42] is a low cost baseline JPEG encoder soft IP and the chip implementation. The features of this design are user defined which supports reconfigurable quantization tables and fully pipelined architecture. The encoder presented in [42] consists of 2-D DCT, quantizer and

Huffman coder. The 2-D DCT module is implemented using row-column decomposition method and the quantization is implemented using a Look-up table based method. Even for this design only the hardware cost of the quantization unit is considered. As in this design the same quantization scheme is used for both VC-1 and MPEG-2, in Table 5.6 hardware cost for both VC-1 and MPEG-2 is identical. Moreover HEVC is the new developing standard and hence the codec for this standard is not yet available. So the cost for standalone HEVC standard is measured by implementing it separately [33]. The cost of the standalone quantization units are added together to find the total estimated cost of six standalone codecs which is 40.95K logic gates. The original cost of the circuit shared quantizer architecture is 20.8K logic gates which saves up to 49.1% than the estimated cost. Table 5.6 demonstrates this whole scenario and Figure 5.1 illustrates the normalized area with the percentage of reduction based on Table 5.6.

### 5.3.2 Estimated area savings in multi-codec decoder

In order to better assess the saving in hardware for the entire decoder, a cost analysis is done as presented in Figure 5.2 where the costs of standalone and shared design are shown.

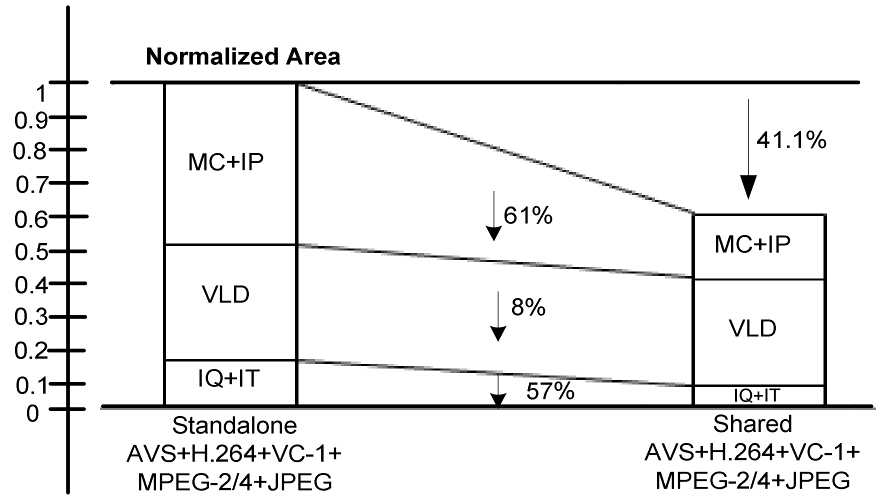

Figure 5.2. Cost reduction map of decoder using the multi quantizer architecture

The cost of a decoder for four codecs (H.264, VC-1, AVS, MPEG2/4) is taken from [31]. This design presents multi format video decoder which integrates AVS JP @ L6.2, H.264 HP @ L4.2, VC-1 AP @ L3 and MPEG-2 MP @ HL in a single chip and features resource sharing, memory management and early - stage acquisition to facilitate cost and bandwidth efficiency. For the applications of broadcasting adoptive error concealment method is used. The cost of JPEG codec is taken from [42] and added with the previous to calculate the total cost for all five codecs. Here, MC is motion compensation, IP is intra prediction, VLD is variable length decoder, IQ is inverse quantization, and IT is inverse transform. To calculate the cost of shared implementation, the cost for MC, IP, and VLD units are taken from the shared design presented in [31]. The cost for IT (for shared design) is taken from our previous work [15], and the cost for IQ is taken from the current work. Thus, it is observed that the shared design (that includes our multi-codec DFQA scheme) is estimated to save overall 41.1% area of a decoder compared to standalone design for five codecs. Note that, Figure 5.2 excludes the emerging HEVC, since as of today the cost of implementing different units for HEVC is not available.

## 5.4 Calculation and comparison of decoding capability

In order to have a better assessment among comparable designs (e.g., minimum support of six codecs), in Table 5.7, the decoding capability of the proposed multi quantization approach is compared with other quantization only designs. While working at maximum capacity on Virtex4 LX60 FPGA, the multi quantizer has the decoding capacity of 90.2 Hz for a full 1080p HD video. At this rate, the maximum achieved frame rate (with 4:2:0 luma-chroma sampling) will be = 187 x 106/(1,920 x 1,080 + 2 x 960 x 540) = 60.1 ≈ 60 fps, which is the highest compared with all. The frame rate of this circuit shared architecture is 45 fps while implemented in Virtex2.

Table 5.7. Comparison of decoding capability of the multi quantizer schemes

| Arch. | Technology | Resolution (1920x1080) | | Resolution (1280x720) | | Comments |
|---|---|---|---|---|---|---|
| | | Time to decode 1 frame (msec) | Frame per second (fps) | Time to decode 1 frame (msec) | Frame per second (fps) | |
| [19] | Virtex2(XC2VP7) | 23.3 | 43 | 10.2 | 97 | Supports only H.264 |
| [24] | XilinxV2P30FF896 | 28.8 | 34 | 12.8 | 78 | Supports only H.264 |
| [25] | Virtex2(XC2VP7) | 33.1 | 30 | 14.7 | 67 | Supports only H.264 |
| Ours | Virtex4(LX60) | 16.6 | 60 | 7.4 | 135 | Supports six codecs |
| Ours | Virtex2(XC2V500) | 21.9 | 45 | 9.7 | 102 | Supports six codecs |

Similarly the decoding capacity of the combined multi transform and multi quantization is also calculated using Virtex4 LX60 FPGA which is reported in Table 5.8.

Table 5.8. Comparison of decoding capability of combined transform-quantizer

| Arch. | Technology | Resolution (1920x1080) | | Resolution (1280x720) | | Comments |
|---|---|---|---|---|---|---|
| | | Time to decode 1 frame (msec) | Frame per second (fps) | Time to decode 1 frame (msec) | Frame per second (fps) | |
| [19] | Virtex2 | 27.8 | 36 | 12.3 | 81 | Supports only H.264 |
| [21] | Virtex2 | N | N | 20 | 49 | Supports only H.264 |
| [22] | StratixII | 31.1 | 32 | 13.8 | 72 | Supports only H.264 |
| [23] | Virtex2 | 27 | 36 | 12 | 83 | Supports only H.264 |
| [26] | Virtex2 | N | N | 17.1 | 58 | Supports only H.264 |
| [27] | Virtex2 | 31.4 | 31 | 14 | 71 | Supports only H.264 |
| Ours | Virtex4 | 15.9 | 62 | 7.1 | 140 | Supports six codecs |
| Ours | Virtex2 | 22.9 | 43 | 10.2 | 98 | Supports six codecs |

‘N’- Does not meet the standard;

So from Table 5.7 and Table 5.8, it is observed that the proposed architecture has the highest decoding capability for both Virtex2 and Virtex4 implementation and can transmit maximum no. of frames per second with minimum amount of time even when implemented in Virtex2.

CHAPTER 6

# CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORKS

## 6.1 Conclusion

In this thesis, a novel generalized 8x8 quantization algorithm and shared hardware architecture which is applicable to six different video codecs: H.264/AVC, AVS, VC-1, MPEG-2/4, MJPEG and HEVC (draft stage) is presented. Since HEVC is a developing standard, the proposed hardware is designed in such way so that it can accommodate any further changes in the final release of HEVC.

Although quantization can be considered as division operation, H.264, AVS and HEVC use multiplication and right shift operation as a replacement of division operation to reduce the random noise introduced due to division. However, the quantization schemes in VC-1, MPEG-2/4 and MJPEG are defined as division operation only (using 8x8 matrices). As a result, it is a challenge to establish a relationship that is general enough to merge all these schemes. After careful observations of all the quantization schemes of different standards a relationship has been found. Moreover, after more intense investigation, it is observed that the quantization in VC-1 is user defined and is similar to that of MPEG-2 standard. After considering all the features, a novel generalized algorithm is developed which is applicable to all six standards.

The hardware of the multi quantizer unit has a maximum operational frequency of 187.3MHz with a decoding capability of 60 fps and 135 fps for full HD resolution (1920x1080) and HD resolution (1280x720) respectively when implemented in Xilinx Virtex4 LX60 FPGA.

All the comparisons show that the proposed hardware can operate at a higher operational frequency with comparable hardware count. The estimated cost of the multi quantizer as well as the multi codec decoder demonstrates a significant amount of area savings. All the performance analysis shows that the proposed design satisfies the requirement of all six codecs and achieves competitive decoding capability. Overall, the architecture is suitable for real-time application and low-cost implementation in modern multi-codec systems.

## 6.2 Suggestions for future works

In this section some suggestions for future works are presented. These are as follows:

I. The multi quantizer unit presented in this thesis can compute 8x8 quantization operation. However the design can be extended to support 4x4 quantization schemes of different standards.

II. The current version of the multi quantizer unit can compute the quantization of six different standards. However the inverse quantization scheme is not developed. Hence the inverse quantization schemes and parameters of different standards can be observed to find a generalized inverse quantization algorithm to design a circuit shared inverse multi quantizer architecture. Then the multi quantizer unit presented in this work can be merged with the inverse multi quantizer unit to develop an I/Q (quantization and inverse quantization) pair for different standards.

III. The presented hardware architecture is an area optimized architecture with one shared multiplier which results a much smaller “foot print”. The hardware architecture can be modified for speed optimization by adding multiple quantizer units. For example the architecture presented here has one quantizer unit to support 8x8 quantization operation. The throughput can be boosted up to the maximum range by replicating 64 quantizer units for 8x8 = 64 coefficients. Hence by applying replications and parallelisms the performance of the multi quantizer unit can be improved to the maximum level.

IV. In this work, a sharing algorithm is developed and implemented for the quantization unit of multiple video codecs. Similar approach can be taken into account to implement other units, such as motion compensation, deblocking filter of different modern video codecs to save area and enhance performance. Finally a full multi codec system can be developed.

## REFERENCES

[1] CCITT recommendation T.81, digital compression and coding continuous-tone still images (1992)

[2] ISO/IEC 13818-2:1995: Information technology—generic coding of moving pictures and associated audio information: Video

[3] Standard for television: VC-1 compressed video bitstream format and decoding process, SMPTE 421 M (2006)

[4] Recommendation ITU-T H.264, ISO/IEC 14496-10:2003, Advanced Video Coding for Generic Audio-Visual Services, 2003

[5] GB/T 20090.1 Information technology—advanced coding of audio and video—Part 1: system, Chinese AVS standard

[6] Meeting report of the sixth meeting of the Joint Collaborative Team on Video Coding (JCT-VC), Available [Online] http://wftp3.itu.int-/av-arch/jctvc-site/

[7] J. B. Lee and H. Kalva, *The VC-1 and H.264 Video Compression Standards for Broadband Video Services*, vol. 32, 1st edn. Springer, Florida, 2008.

[8] D. T. Vo and T. Q. Nguyen, "Quality Enhancement for Motion JPEG Using Temporal Redundancies", *IEEE trans. on Circuits and Systems for Video Technology,* vol. 18, no. 5, pp. 609-619, May 2008.

[9] S. Srinivasan, P. Hsu, T. Holcomb, K. Mukerjee, S. L. Regunathan, B. Lin, J. Liang, M. C. Lee and J. R. Corbera, "Windows Media Video 9: Overview and Applications", *Research Article Signal Process. : Img. Communicaation*, vol. 19, no. 9, pp. 851-875, Oct. 2004.

[10] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer and T. Wedi, "Video Coding with H.264/AVC: Tools, Performance, and Complexity", *Circuits and Systems Magazine IEEE*, Vol. 4, Issue 1, pp. 7-28, 2004.

[11] L. Yu, F. Yi, J. Dong and C. Zhang, "Overview of AVS-Video: Tools, Performance and Complexity", *J.* of *Visual Communications and Image Process.*, vol. 5960, pp. 679-690, 2005.

[12] B. Waggoner, *Compression for Great Video and Audio,* 2nd edn. Elsevier, 2009.

[13] A. Vetro, C. Christopoulos, S. Huifang, E. Mitsubishi, M. Hill, "Video Transcoding Architectures and Techniques: an Overview", *IEEE Signal Process. Magz.*, vol. 20, pp. 18–29, Mar. 2003.

[14] I. Ahmad, X. Wei, Y. Sun, Y. Zhang, “Video Transcoding: an Overview of Various Techniques and Research Issues”, *IEEE Trans. Multimedia*, vol. 7, pp. 793–804, Oct. 2005.

[15] Khan A. Wahid, Md. Martuza, M. Das and C. McCrosky, “Efficient Hardware Implementation of 8x8 Integer Cosine Transforms for Multiple Video Codecs”, *J. of Real-Time Image Processing, Springer*, 8 pages, doi: 10.1007/s11554-011-0209-6, in press (available online)., 2011.

[16] S. Lee and K. Cho, "Architecture of Transform Circuit for Video Decoder Supporting Multiple Standards", *Electron. Lett.*, vol. 44, no. 4, pp. 274-275, 2008.

[17] S. Kim, H. Chang, S. Lee and K. Cho, "VLSI Design to Unify IDCT and IQ Circuit for Multistandard Video Decoder", *in: Proc. ISIC Int. Symposium on Prototyping,* pp. 328-331, 2008.

[18] S. Lee and K. Cho, "Circuit Implementation for Transform and Quantization Operations of H.264/MPEG-4/VC-1 Video Decoder", *in Proc. Int. Conf. on Design and Technology of Integrated Systems in Nanoscale Era,* pp. 102-107, 2007.

[19] R. C. Kordasiewicz and S. Shirani, “ASIC and FPGA Implementations of H.264 DCT and Quantization Blocks”, *in proc. IEEE Int. Conf. on Image Process.*, pp. III - 1020-3, Sept. 2005.

[20] J. S. Park and T. Ogunfunmi, “A New Hardware Implementation of the H.264 8x8 Transform and Quantization”, *in proc. IEEE Int. Conf. on Acoustics, Speech and Signal Process.*, pp. 585-588, April, 2009.

[21] I. Amer, W. Badawy and G. Jullien, “A High-Performance Hardware Implementation of the H.264 Simplified 8X8 Transformation and Quantization”, *in proc. IEEE Int. Conf. on Acoustics, Speech and Sig. Process.*, vol. 2, pp. 1137-1140, Mar. 2005.

[22] G. Pastuszak, “Transforms and Quantization in the High-Throughput H.264/AVC Encoder Based on Advanced Mode Selection”, *in proc. IEEE Computer Society Annual Symposium on VLSI*, pp. 203-208, April, 2008.

[23] X. T. Tran and V. H. Tran, “Cost-Efficient 130nm TSMC Forward Transform and Quantization for H.264/AVC Encoders”, *in proc. IEEE 14th Int. Symp. on Design and Diagnostics of Elect. Circuits and Systs.*, pp. 47-52, April, 2011.

[24] R. Husemann, M. Majolo, V. Guimaraes, A. Susin, V. Roesler and J. V. Lima, “Hardware Integrated Quantization Solution for Improvement of Computational H.264 Encoder Module”, *in proc. IEEE/IFIP VLSI System on Chip Conf.*, pp. 316-321, Sept. 2010.

[25] R. Kordasiewicz and S. Shirani, “Hardware Implementation of the Optimized Transform and Quantization Blocks of H.264”, *in proc. Canadian Conf. on Elect. and Computer*, vol. 2, pp. 943-946, May, 2004.

[26] O. Tasdizen and I. Hamzaoglu, “A High Performance and Low Cost Hardware Architecture for H.264 Transform and Quantization Algorithms”, *in proc. 13th European Signal Process. Conf.*, pages 4–8, Sept. 2005.

[27] C. P. Fan and Y. L. Cheng, “FPGA Implementations of Low Latency and High Throughput 4x4 Block Texture Coding Processor for H.264/AVC”, *J. of the Chinese Institute of Engineers*, vol. 32, no. 1, pp. 33–44, 2009.

[28] K. Zhang, Y. Zhu and L. Yu, “Area-Efficient Quantization Architecture with Zero-Prediction Method for AVS Encoders”, *in proc. Picture Coding Symp.*, 4 pages, Nov. 2007.

[29] K. Suh, K. Y. Min, K. Kim, J. S. Koh and J. W. Chong, “A Design of DPCM Hybrid Coding Loop Using Single 1-D DCT in MPEG-2 Video Encoder” , *in proc. IEEE Int. Symp. on Circuits and Systs.*, vol. 4, pp. 279-282, July, 1999.

[30] H. Osman, W. Mahjoup, A. Nabih, and G. M. Aly, “JPEG Encoder for Low-Cost FPGAs”, *in proc. Int. Conf. on Comp. Eng. and Syst.*, pp. 406-411, Nov. 2007.

[31] C. C. Ju, Y. C. Chang, C. Y. Cheng, C. M. Wang, H. M. Lin, C. C. Chen, F. Chiu and S. J. Wang, “A Full HD 60fps AVS/H.264/VC-1/MPEG-2 Video Decoder for Digital Home Applications”, *in proc. Int. Symp. on VLSI Design, Automation and Test (VLSI-DAT)*, pp. 1-4, April, 2011.

[32] Silicon Image Inc., http://www.siliconimage.com/products/index.aspx, Available [Online], May, 2011.

[33] A. Fuldseth, G. Bjøntegaard, M. Budagavi and V. Sze, “Core Transform Design for HEVC”, JCTVC-G495, Geneva, Nov. 2011.

[34] M. Martuza and Khan Wahid, “Low Cost Design of a Hybrid Architecture of Integer Inverse DCT for AVS, H.264, VC-1 and HEVC”, *VLSI Design, Special Issue on VLSI Circuits, Systems, and Architectures for Advanced Image and Video Compression Standards, Hindawi Publishing*, vol. 2012, Article ID: 242989, 11 pages, in press.

[35] B. Prasad and W. Badawy, “Assessment of Hardware vs Software Implementations for Video Microscopy”, *in proc. Biomedical Circuits and Systs. Conf.*, pp. 150-153, 2006.

[36] J. Liu and D. Liang, “A Survey of FPGA-Based Hardware Implementation of ANNs”, *Int. Conf. on Neural Networks and Brain*, vol. 2, pp. 915-918, Oct. 2005.

[37] *Course content of CME426: Multimedia* [Online]. Available: http://engrwww. usask.ca /classes/CME/462.

[38] Z. Khan and A. B. Mansoor, “An Analysis of Quality Factor on Image Steganalysis”, *in proc. Informatics and Systems, 7th Int. Conf.*, pp. 1-5, March 2010.

[39] E.Alshina, A.Alshin, K. Kim and P. Topiwala, “Full-Factorized Core Transform Proposal by Samsung Electronics, Ltd/FastVDO”, JCTVC-F251, Torino, July 2011.

[40] A. Fuldseth, G. Bjøntegaard, M. Budagavi and V. Sze, “Core Transform Design for HEVC”, JCTVC-F446, Torino, July 2011.

[41] A. Fuldseth, G. Bjøntegaard, M. Budagavi and V. Sze, “Core Transform Design for HEVC”, JCTVC-E243, Geneva, Mar. 2011.

[42] C. J. Lian, L. G. Chen, H. C. Chang and Y. C. Chang, “Design and Implementation of JPEG Encoder IP Core”, *in proc. Asia and South Pacific Design Automation Conf.*, pp. 29-30, 2001.

[43] D. Salomon and G. Motta, *Handbook of Data Compression*, 5th edn. Springer, New York, 2010.

[44] S. Gordon, D. Marpe, and T.Wiegand, Simplified Use of 8x8 Transform - Proposal. *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG*, doc. JVT-J029, Waikoloa, USA, Dec. 2003.

[45] S. Kwon, A. Tamhankar and K.R. Rao, “Overview of H.264/MPEG-4 part 10”, *J. Visual Communication and Image Representation, Special issue on Emerging H.264/AVC video coding standard*, vol. 17, pp. 186–216, Apr. 2006.