International Journal of Reconfigurable and Embedded Systems (IJRES)
Vol. 2, No. 3, November 2013, pp. 116~121
ISSN: 2089-4864

An Efficient Implementation of the Entire Transforms in the H.264/AVC Encoder using VHDL

Farhad Rad*, Ali Broumandnia**
* Departement of Computer Engineering,College of Graduate Studies, Science and Research Branch of Kohgiluyeh and Boyer-ahmad, Islamic Azad University, Yasouj, Iran
** Departement of Computer Engineering, Science and Research Branch of Tehran, Islamic Azad University, Tehran, Iran

Article Info

Article history:

Received Aug 2, 2013
Revised Oct 14, 2013
Accepted Oct 28, 2013

Keyword:

Discrete Cosine Transform
H.264 Encoder
Hadamard Transform
Integer DCT
VHDL

ABSTRACT

The H.264/AVC standard achieves remarkable higher compression performance than the previous MPEG and H.26X standards. One of the computationally intensive units in the MPEG and H.26X video coding families is the Discrete Cosine Transform (DCT). In this paper, we propose an efficient implementation of the DCT, inverse DCTs and the Hadamard transforms in the H.264/AVC encoder using VHDL. The synthesis results indicate that our implementation of the entire transforms achieves lower power, delay and area consumption compared to the existing architectures in the H.264/AVC encoder.

Corresponding Author:

Farhad Rad,
Departement of Computer Engineering,College of Graduate Studies,
Science and Research Branch of Kohgiluyeh and Boyer-ahmad,
Islamic Azad University, Yasouj, Iran.
Email: F_radl@hotmail.com

1. INTRODUCTION

Compression is the process of reducing the size of the data sent, thereby, reducing the bandwidth required for the digital representation of a signal. Many inexpensive video and audio applications are made possible by the compression of signals. Compression technology can result in reduced transmission time due to less data being transmitted. It also decreases the storage requirements because there is less data. However, signal quality, implementation complexity, and the introduction of communication delay are potential negative factors that should be considered when choosing compression technology [3]. The H.264/AVC standard [1] achieves remarkable higher compression performance than the previous MPEG and H.26X standards. The higher performance in H.264/AVC is due to various modifications in different coding stages and most of these modifications impose high computational load to the H.264/AVC codec [5]-[7]. One of the computationally intensive units in the MPEG and H.26X video coding families is the Discrete Cosine Transform (DCT). Hence, the architecture exploration of this unit is even attractive for the pre-H.264/AVC standards and there are proposals for hardware architectures to realize this unit from a long time ago [8] and it is still continuing [9]-[10]. The initial version of H.264/AVC standard supported only 4×4 integer DCT. The number of operations for computation of an 8×8 or 4×4 Integer DCT is not very high but since in high profiles theses transforms should be applied to the entire 8×8 or 4×4 blocks in a frame, it will result in a huge computational load and makes the integer discrete cosine transform among main computationally intensive stages in the H.264 encoder. Consequently, the hardware implementation of the integer DCT transform attracted more attention and a number of solutions have been published for hardware implementation of

Integer DCT in the H.264/AVC standard [11]-[12]. In this paper, we proposed an efficient implementation of the entire transforms in the H.264/AVC Encoder using VHDL.

Since the encoding loop of the H.264/AVC standard requires carrying out all the forward and inverse transforms, the proposed implementation is a very powerful accelerator for the H.264/AVC encoder. The synthesis results indicate that our implementation of the DCT, inverse DCTs and the Hadamard transforms in the H.264/AVC encoder achieves lower power, delay and area consumption compared to the existing architectures in the H.264/AVC encoder.

The rest of the paper is organized as follows. In section 2 we provide a brief overview of the transforms in the H.264/AVC standard and discuss the initial hardware implementation. The proposed implementation of the entire transforms in the H.264/AVC standard is explained in section 3. VHDL and synthesis results for the given architecture and comparison with the primal architecture are presented in section 4. Finally, Section 5 concludes the paper.

2. THE H.264/AVC STANDARD

In the H.264/AVC standard the forward and the inverse integer DCT are defined respectively in (1) and (2) as:

$$Y = AXA^T \Rightarrow Y = C_f X C_f^T \otimes E_f \quad (1)$$

$$X = A^T Y A \Rightarrow X = C_i^T (Y \otimes E_i) C_i \quad (2)$$

Where $C_f X C_f^T$ and $C_i^T (Y \otimes E_i) C_i$ are called ‘’core‟ transforms [2]. The C_f and C_i matrices given in (3) indicate the ‘’core‟ transform matrix of the forward and inverse 4×4 integer DCT in the H.264/AVC standard, respectively [zargari, malvar].

$$C_f = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \qquad C_i = \begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{bmatrix} \quad (3)$$

H.264 is unique that it employees this purely integer spatial transform as opposed to the usual floating point 8x8 DCT specified with rounding error tolerance as used in earlier standards [2]. The small shape helps to reduce the blocking and ringing artefacts, while the precise integer specification eliminates any mismatch between the encoder and decoder in the inverse transform. The multiplications by 1/2 in the inverse transform can be implemented by the sign preserving1-bit right shifts thus reducing the complexity. Hadamard transform is another 2D transform which is used in the H.264/AVC standard and its ‘’core‟ transform matrix is:

$$H_{4\times4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \quad (4)$$

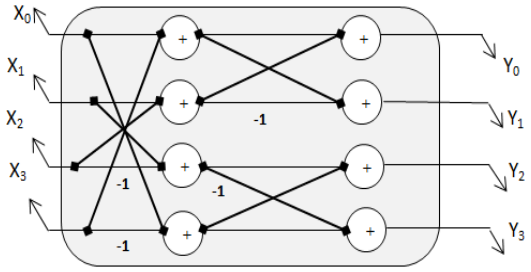

Figure 1. Quick hardware for 4×4 Hadamard Transform condition

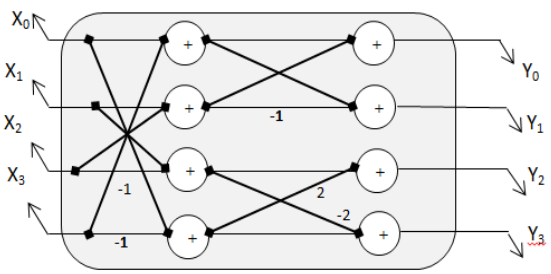

Figure 2. Quick hardware for forward 4×4 Integer DCT

Since $H^{T}_{4\times4} = H_{4\times4}$, the hardware implementation given in Figure 1 can be used for both forward and inverse Hadamard transforms. Figure 2 shows an quick hardware realization for 4×4 forward integer transform using adders and shifters and Figure 3 indicates a quick hardware realization for 4×4 inverse integer transform [2].

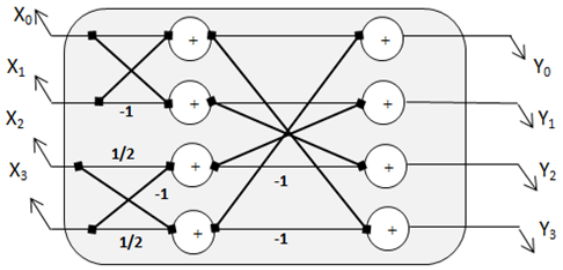

Figure 3. Quick hardware forinverse 4×4 Integer DCT

3. THE PROPOSED IMPLEMENTATION

This section introduces the proposed implementation of the 4×4 forward transform and 4×4 inverse transform adopted by the H.264 standard. 'Core' transform is a two dimensional transform, which can be decomposed into two one dimensional transforms. The first one dimensional transform is applied to the rows of the input pixels and the second one dimensional transform is applied to the columns of the one dimensional transform coefficients of the first stage. The proposed architecture uses 4×4 parallel input blocks; a block diagram of the architecture is shown in Figure 4. This block consists of four cascaded sub-blocks. The control unit is used to control add and shift operations. The register bank stores these outputs for the next four clock cycles.

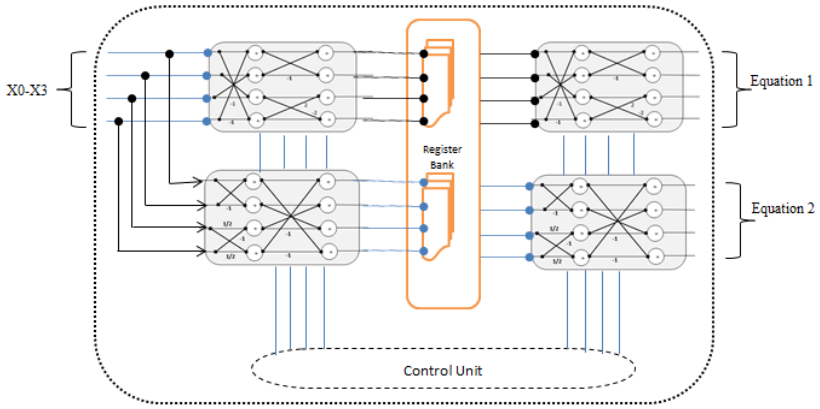

Figure 4. Architecture of core forward and inverse integer transforms.

4. VHDL AND SYNTHESIS RESULT

VHDL simulation results for forward/inverse 4×4 Integer DCT and Hadamard transform are depicted in figure 5-10. After successful simulation of working of different blocks of H.264 encoder (Integer DCT, Inverse DCT, Hadamard transform), we then proceeded for synthesis of code on Xilinx.ISE.DESIGN.SUITE.v12.3 and Synopsis.And also we have compared our synthesis results with the performance of existing implementations [4]. Comparison shows that power consumption and delay are lower than other implementations. Also, the area of our chip is much smaller than others. Summary of the synthesis results are shown in Tables 1, 2 and 3.

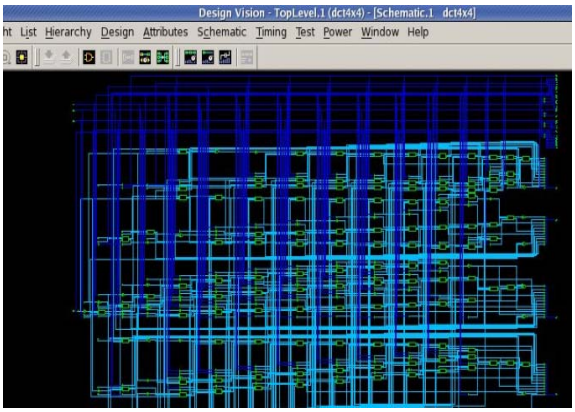

Figure 5. Simulation Result of Quick DCT4*4 in VHDL

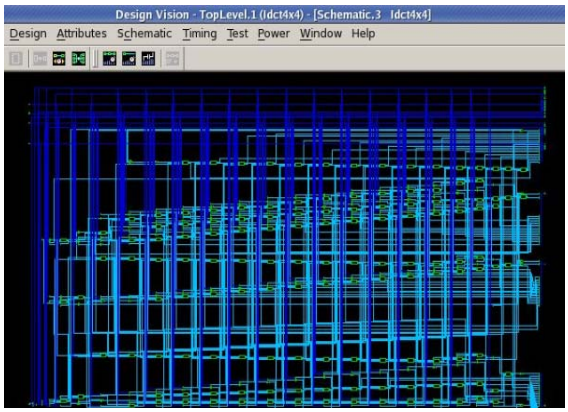

Figure 6. Simulation Result of Quick IDCT4*4 in VHDL

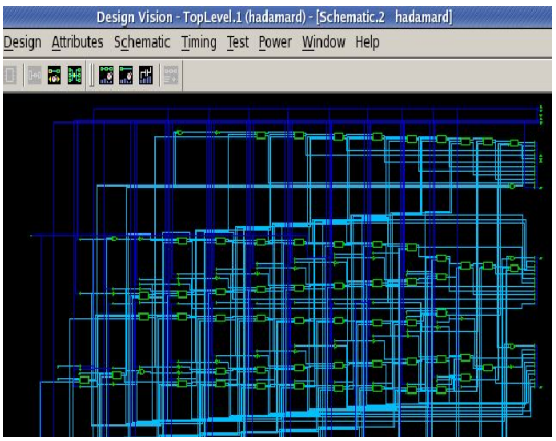

Figure 7. Simulation Result of Hadamard 4*4 in VHDL

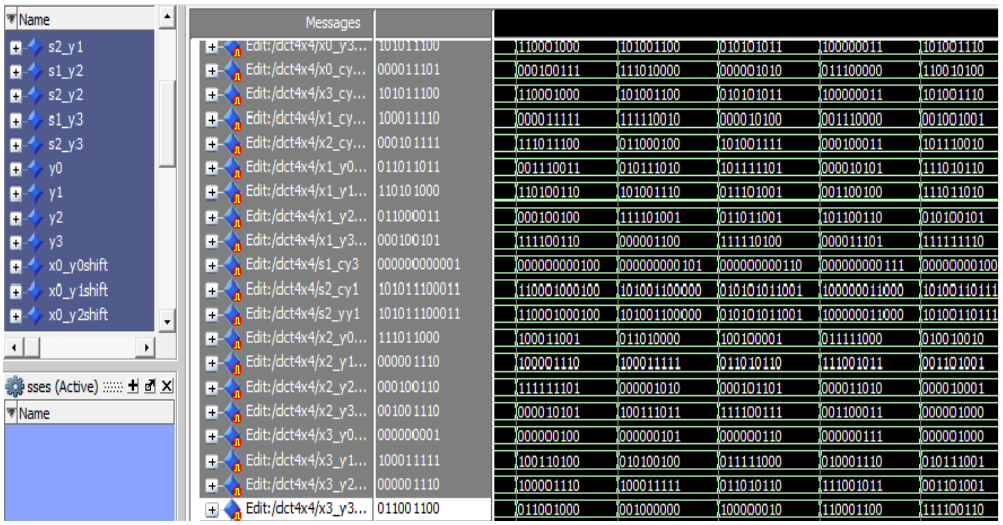

Figure 8. DCT 4*4 Test in VHDL

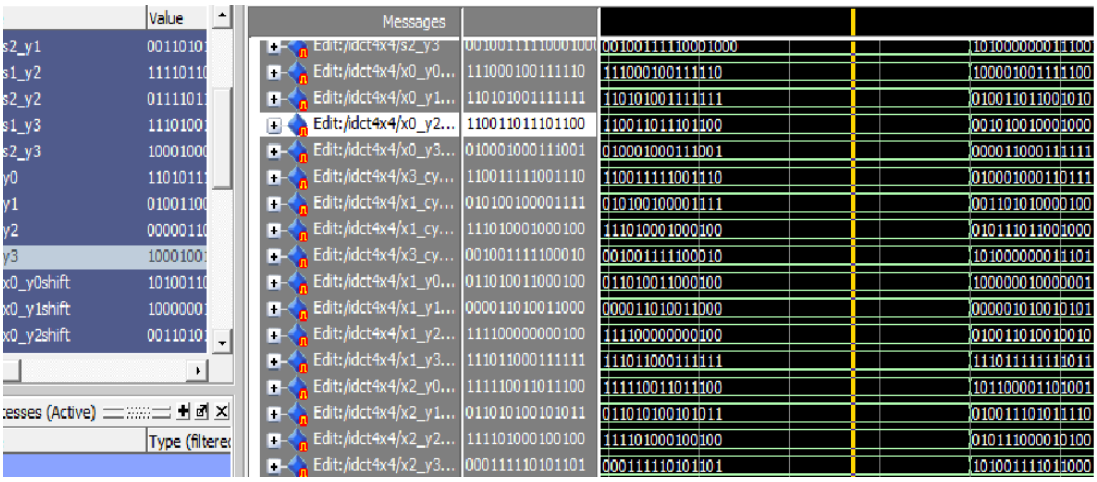

Figure 9. IDCT 4*4 Test in VHDL

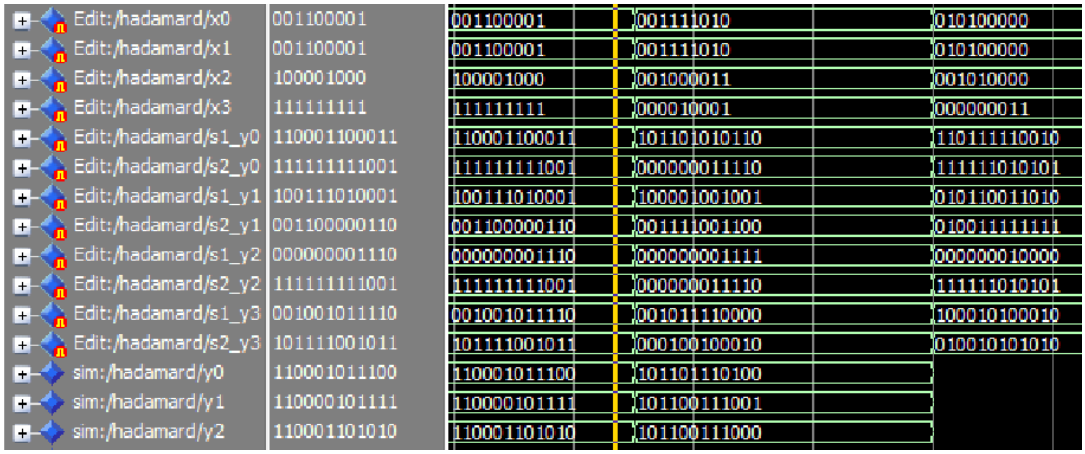

Figure 10. Hadamard 4*4 test in VHDL

Table 1. Synthesis Results of DCT 4*4

Implementation	Power(mW)	Delay(ns)	Area(MicroM2)
Existing DCT	4.1494	2.36	10812.413114
Proposed DCT	3.2174	2.06	7269.419728

Table 2. Synthesis Results of IDCT 4*4

Implementation	Power(mW)	Delay(ns)	Area(MicroM2)
Existing IDCT	7.0668	2.76	17278.790363
Proposed IDCT	5.7302	2.76	12943.068996

Table 3. Synthesis Results of Hadamard 4*4

Implementation	Power(mW)	Delay(ns)	Area(MicroM2)
Existing Hadamard	3.1193	2.55	7223.282946
Proposed Hadamard	3.0934	2.44	7223.161031

Table 4. Synthesis Result Comparison of Core Transform with Existing Implementation

	This work	Existing implementation [4]
No. of flip flops	58	65
No. of Slices	159	167
Max. freq	193MHz	185MHz

5. CONCLUSION

In this paper, an efficient implementation of the core processors of H.264 video encoder is proposed. The 4x4 integer transform used is significantly simpler and faster than the 8×8 DCT used in MPEG 2. This transform is a scaled integer approximation to the DCT, which allows computation of the direct or inverse transform with just additions and a minimal number of shifts, but no multiplications. Synthesis results indicate that our implementation, which realizes the entire transforms in the H.264/AVC standard, can process lower power, delay and area consumption compared to the existing architectures [4], which implement a number of the transforms in H.264/AVC.Also, comparison shows that our design achieved higher frequency.

REFERENCES

[1] ITU-T Rec. H.264 / ISO/IEC 11496-10.*Advanced video coding for generic audiovisual services.* 2005.
[2] H Malvar, A Hallapuro, M Karczewicz, L Kerofsky. Low-complexity transform and quantization in H.264/AVC. *IEEE Trans. Circuit syst. Video Techno,.* 2003; 3(7): 598-603.
[3] F Zargari, S Ghorbani. *A Hardware Sharing Architecture for Implementing the entire Transforms in H.264/AVC Video Coding Standard.* 15th IEEE International Symposium on Consumer Electronics (ISCE2011). 2011: 14-17.
[4] S Kumar, S Popuri, R Pankaj. Design and Implementation of Transform and Quantization Blocks of H.264 Video Encoder using VHDL and MATLAB. *International Journal of Emerging Trends in Engineering and Development,* 2013; 3(2).
[5] L Li, Y Song, S Li, T Ikenaga, S Goto. A Hardware Architecture of CABAC Encoding and Decoding with Dynamic Pipeline for H.264/AVC. *Journal of Signal Processing Systems.* 2008; 50(1): 81-95.
[6] M Kthiri, HN Loukil, A Atitallah, P Kadionik, D Dallet, et al. FPGA architecture of the LDPS Motion Estimation for H.264/AVC Video Coding. *Journal of Signal Processing Systems, Online First™*. 2011.
[7] GA Ruiz, JA Michell. An Efficient VLSI Architecture of Fractional Motion Estimation in H.264 for HDTV. *Journal of Signal Processing Systems*. 2011; 62(3): 443-457.
[8] W Liebsch. *Parallel architecture for VLSI implementation of a 2-dimensional discrete cosine transform for image coding.* Third International Conference on Image Processing and its Applications. 1989: 609–612.
[9] J Park, K Roy. A Low Complexity Reconfigurable DCT Architecture to Trade off Image Quality for Power Consumption. *Journal of Signal Processing Systems.* 2008; 53(3): 399-410.
[10] Z Wu, J Sha, Z Wang, L Li, M Gao. An Improved Scaled DCT Architecture. *IEEE Transactions on Consumer Electronics,* 2009; 55(2): 685-689.
[11] JS Park, T Ogunfunmi. *A New Hardware Implementation of the H.264 8×8 Transform and Quantization*. Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing. 2009: 585-588.
[12] R Korah, JRP Perinbam. FPGA Implementation of Integer Transform and Quantizer for H.264 Encoder. Journal of Signal Processing Systems. 2008; 53(3): 261-269.

BIOGRAPHIES OF AUTHORS

Farhad Rad received B.Sc. in Computer Hardware Engineering from Shiraz University in Shiraz, Iran in 2005, M.Sc. in Computer Hardware Engineering fromIran University of Science and technology in Tehran, Iran in 2008, andPh.D. Student in Hardware Engineering in Islamic Azad University, Science and Research Branch in Tehran, Iran. His research interests include System-on-Chip design and verification, embedded systems, VLSI systems/circuits design for multimedia application.

Ali Broumandnia was born in Esfahan, Iran in 1967. He received B.Sc. inComputer Hardware Engineering from Esfahan University of Technologyin Esfahan, Iran in 1992, M.Sc. in Computer Hardware Engineering fromIran University of Science and Technology in Tehran, Iran in 1995, andPh.D. degree in Computer Engineering from Islamic Azad University,Science and Research Branch in Tehran, Iran, in 2006. He is interested in pattern recognition, documentimage analysis, and wavelet analysis.