A NOVEL PARALLEL 4X4 TRANSFORM AND INVERSE TRANSFORM ARCHITECTURE FOR H.264

Tiejun Li,
School of Computer Science, National University of Defense Technology, ChangSha, China
tj_li@sohu.com

Sikun Li,
School of Computer Science, National University of Defense Technology, ChangSha, China
lisikun@263.net.cn

DOI: 10.36724/2664-066X-2021-7-5-31-35

ABSTRACT

The H.264 video coding standard provides a compression gain of 1.5 x 2.0 x over H.263 and MPEG-4 simple profile. One of the major differences between H.263 and H.264 is the transform coding. The transforms of H.264 employ only integer arithmetic operations such as additions and shifts without multiplications, with coefficients and scaling factors that allow for 16-bit arithmetic computation on first-level transforms. The integer transforms also solve the mismatch problem like H.263. These changes lead to a significant complexity reduction, but with only less than 0.02 dB decrease in PSNR. A novel parallel 4x4 multiple transforms architecture for H.264 is presented in this paper. This architecture is based on Wallace trees and can process 4 inputs in parallel. This architecture has been designed and synthesized in SMIC 0.18um technology. The result shows that this architecture can achieves above 1,200M pixels/sec throughout and consume only 5625 gates. The timing-area property of this architecture is improved compared with the previous architecture.

KEYWORDS: *Architecture, Transform, H.264, Wallace Tree.*

This work is supported by National Nature of Science Foundation of China (90207019) and "Hi-Tech Research & Development Program (863)" of China

The article is reworked from unpublished 2nd IEEE International Conference on Circuits and Systems for Communications (ICCSC) materials.

INTRODUCTION

The H.264 video coding standard provides a compression gain of 1.5 x – 2.0 x over H.263 [1] and MPEG-4 simple profile [2]. One of the major differences between H.263 and H.264 is the transform coding [3, 4]. The transforms of H.264 employ only integer arithmetic operations such as additions and shifts without multiplications, with coefficients and scaling factors that allow for 16-bit arithmetic computation on first-level transforms. The integer transforms also solve the mismatch problem like H.263. These changes lead to a significant complexity reduction, but with only less than 0.02 dB decrease in PSNR [4, 5].

Although the computation complexity of the transforms in the H.264 standard is less than H.263, hardware implementation of the transform coding is required in both dedicated and platform codec systems to accelerate the processing and alleviate the loading [6]. Hence, Wang [7] proposed a parallel transform architecture for H.264, which can process 4 pixels per cycle and achieve 360M pixels/sec at 80MHz.

The Wallace tree was proposed by C. S. Wallace in 1964 [8]. This method can be used to sum up all the bits of the partial product in each column and is employed by many fast multipliers [9, 10]. Based on Wallace trees, we propose a new parallel 4x4 transform and inverse transform architecture for H.264 in this paper. The new architecture has the following characteristics:

- It takes Wallace trees instead of adders of four operands and can decrease the length of critical paths and gates count.
- Its four parallel datapaths are more independent and need fewer multiplexers than Wang's.
- The subtracters needed by H.264 transforms are merged in Wallace Trees.

Experimental results show that the area of our architecture decreases about 10% and the critical path delay decreases about 20% compared with Wang's [7].

This paper is organized as followed: Sec.2 introduces the transform coding in H.264. Sec.3 presents the architecture based on adders proposed by Wang[7]. Sec. 4 gives the parallel 4x4 transform and inverse transform architecture based Wallace trees proposed by us. Sec.5 implements and compares the proposed architecture and Wang's.

TRANSFORM CODING IN H.264

The video coding layer of H.264 is similar in spirit to other standards such as H.263. It consists of a hybrid of temporal and spatial prediction, in conjunction with transform coding. Figure 1 shows a block diagram of the video coding layer for a macroblock(MB). The spatial prediction is also called intra prediction. There are two types of intra prediction in H.264. One is 4x4 intra prediction and the other is 16x16 intra prediction. The temporal prediction is the multiple reference frame and variable block size motion estimation. Besides predictions and transforms techniques, H.264 also employ other advanced video encode technique, such as context adaptive variable length coding (CAVLC), Context-based Adaptive Binary Arithmetic Coding (CABAC) and deblocking filter.

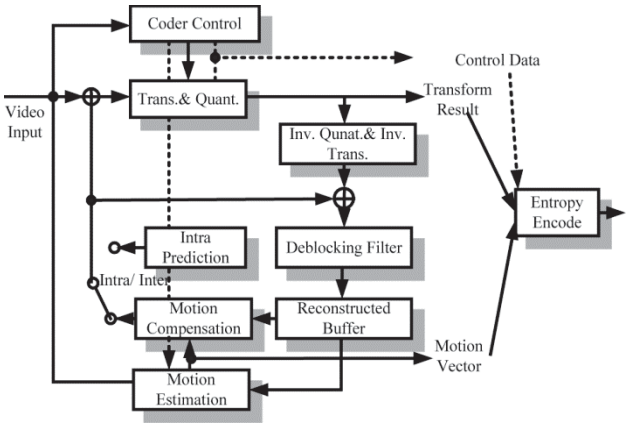

Fig. 1. Block diagram of H.264 encoding flow

The residual MB is coded as Figure 2. It is divided into sixteen 4x4 blocks for luminance and 4 blocks for chrominance. If the 16x16 intra prediction mode is chosen for the MB, the DC items of the luminance are extracted to form a 4x4 block. Hadamard transform is then applied on it.

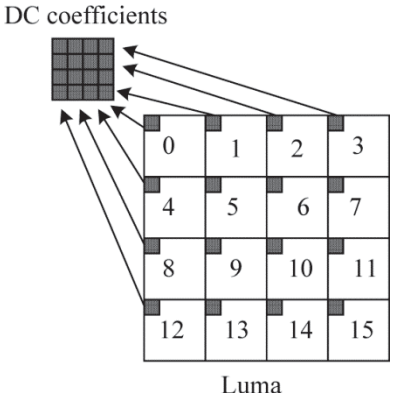

Fig. 2. Residual coding order of H.264

Just like 8x8 DCT, we can implement the 2-D 4x4 integer DCT in this paper by row-column decomposition techniques. There are two types of 4x4 transforms for the residual coding. The first one is for luminance residual blocks. The 1-D transform and inverse transform matrices are shown in (1) and (2).

$$\begin{bmatrix} F0 \\ F1 \\ F2 \\ F3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} X0 \\ X1 \\ X2 \\ X3 \end{bmatrix} \quad (1)$$

$$\begin{bmatrix} X'0 \\ X'1 \\ X'2 \\ X'3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{bmatrix} \begin{bmatrix} F'0 \\ F'1 \\ F'2 \\ F'3 \end{bmatrix} \quad (2)$$

The other type of the transform is Hadamard transform. It is applied to the luminance DC coefficients in 16x16 intra prediction mode. The 1-D inverse Hadamard transform is simply the transpose of (3). Because the transform matrix is symmetric, the inverse hadamard transform is the same as the forward transform.

$$\begin{bmatrix} F0 \\ F1 \\ F2 \\ F3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} X0 \\ X1 \\ X2 \\ X3 \end{bmatrix} \quad (3)$$

The three types of transform matrices above only contain 6 coefficients: 1, -1, 2, -2, 1/2 and -1/2, which can be implemented by shifters and adders. The sign of the coefficient at the same position is same. These characters are helpful to simplify the design of the transform architecture.

TRANSFORM ARCHITECTURE BASED ON ADDERS

Wang [7] implemented a 4x4 integer DCT architecture by row-column decomposition techniques. He overlapped three transforms in (1), (2) in a single architecture, which is showed in Figure 3. In Figure 3, all the adders have three inputs, among which one input is not changed and one of the other two inputs is required to be selected by the transform type. For the different transform types, some inputs should be multiplied by corresponding coefficients: 1, -1, 2, - 2, 1/2 or –1/2.

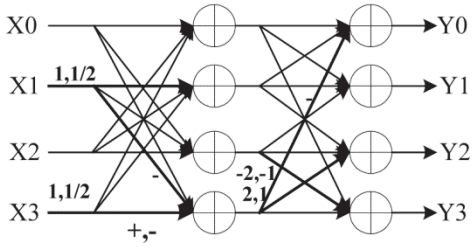

Fig. 3. Architecture for three types transform based adders

PROPOSED ARCHITECTURE

2-D 4x4 Parallel Transform Architecture

We also adopt the classical row-column decomposition method [11, 7] for 2-D 4x4 transform and inverse transform architecture for H.264, as shown in Figure 4.

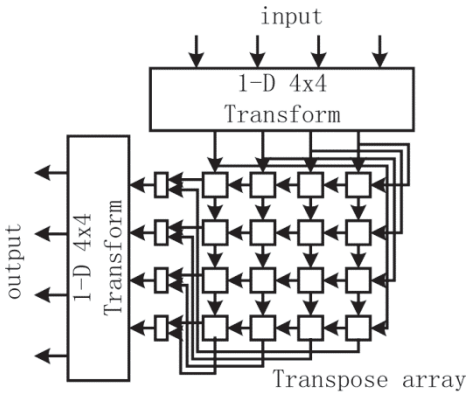

Fig. 4. 4x4 2-D transform architecture

The output of the first 1-D 4x4 transform module on the top of Fig.4 is fed to a transpose array. The transposing result of transpose array is fed to the second 1-D 4x4 transform module as the left transform part of Fig.4. The two transform modules and transpose array form a pipeline structure, which enable the architecture can process 4 pixels per cycle.

The transpose array is a 2-D FIFO array, which can shift along two directions. One FIFO element consists of a two input multiplexer and a register. The multiplexer controls the data flow direction of the FIFO. The first input of the multiplexer is the data from the upper register. The second input of the multiplexer is the data from the right register. If all of the multiplexers select the data from the upper registers, the FIFO array will be shift down. The bottom row of the register array will be outputted to the second 1D transform unit. However, if all of the multiplexers select the data from the right, the FIFO array will be shift left. The direction will be changed every four valid input clocks so that the transpose operation can be done in this register array.

1-D 4x4 Transform Architecture

The 1-D 4x4 transform module is shown in Figure 5, which contains four Wallace trees and eight shift modules.

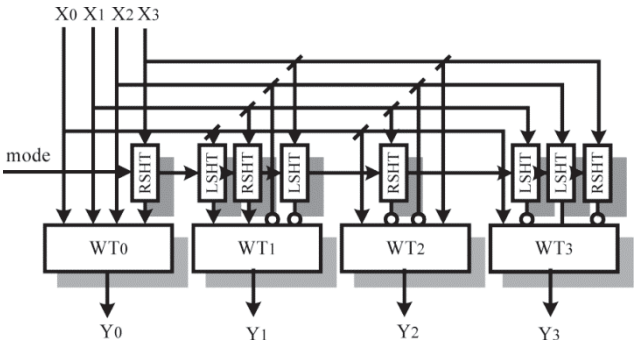

Fig. 5. Architecture for three types transform based Wallace trees

The structure make the best of the characters of 4x4 transforms of H.264 discussed in Sec.2. The four parallel datapaths are independent from each other and intercross little. The core of the datapath is a Wallace tree. Every Wallace tree implements a 4x1 transform. The inputs of the Wallace trees are independent and do not need to be selected by multiplexers. In order to implement the multiplying of 2 and 1/2, two type shift modules are designed: right shifts (RSHT) and left shifts (LSHT). The sign 'o' before some inputs of Wallace trees are designed for subtracting operations.

The shift modules are controlled by the transform mode to decide whether or not to shift and introduce only a delay of one level gate. If the transform mode is the 4x4 transform for luminance residual blocks as (1), LSHTs are selected. If the transform mode is the 4x4 inverse transform for luminance residual blocks as (2), RSHTs are selected. Otherwise, all shifts are all deselected.

Wallace Trees Architecture

The structure of a Wallace tree is shown in Figure 6. In our architecture, Wallace trees are used to substitute for the adder trees of four operands.

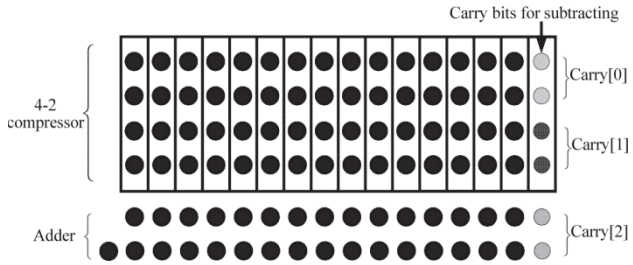

Fig. 6. Wallace tree supporting subtracting

The Wallace trees offer three advantages compared with adder trees. Firstly, a single 4-2 compressor contains only 6 gates and includes a critical path with the maximal delay of three XORs [9, 10], which will decrease the cost of the area and timing. Secondly, the structure of Wallace trees is more regular than adder trees' and is suitable for the placement and routing. Finally, the subtracting needed by H.264 can easily be merged into Wallace trees while introducing a little of gates.

A subtracting can be looked as an addition of the minuend and the negative of the subtrahend. The negative of the subtrahend can be calculated by complementing each bit and then adding 1. In Figure 5, the sign 'o' before the inputs of Wallace trees denotes the operation of complementing. The operations of adding 1 are postponed into Wallace trees.

By observation of Figure 6, we can find that there is one additional bit in the compressor structure and adder, which can generate four types of carrying value (0,1,2 and 3) to add to original calculating of Wallace trees.

This structure consumes a little of gates and introduces the single delay of INVERT (viz. complementing) to implement three possible types of subtracting in 4x4 transforms of H.264.

Bit Width Design

Our proposed architecture should meet the bit width requirements of three types of transform. Both the specification of H.264 [3] and the analysis [7] guarantee that the 16 bits arithmetic is enough, so the datapaths and transpose array in our architecture are implemented with 16 bits.

IMPLEMENTATIONS AND COMPARISION

The proposed architecture in this paper is designed and simulated by Verilog HDL. The logic is synthesized by Synopsys' DC with the Verisilicon standard cell library [12] on SIMIC 0.18um 1P5M process. In order to compare the implementation results to Wang's architecture fairly, the version of Wang's multi-transforms architecture on the same technology and constraint as us is also implemented.

The other individual architectures proposed by Wang are not implement, because the multiple transforms are always required by H.264 but not any individual transform. The gate count and delay are listed in Table 1.

Table 1

Implementation results of our proposed parallel and Wang's

Architecture	Technology	Critical path delay (ns)	Area (gates)
Wang's[7]	TSMC 0.35um	11.35	6538
Wang's	SIMIC 0.18um	5.35	6305
Proposed	SIMIC 0.18um	4.27	5625

CONCLUSIONS

In this paper, we propose a novel parallel 4x4 multiple transforms architecture for H.264. This architecture supports Forward/Inverse transforms for luma residual blocks and DC coefficients used in H.264 and can process 4 inputs in parallel. We have mapped the architecture to SMIC 0.18um technology.

The architecture can run on a system clock of more than 300MHz and achieve 1,200M pixels/sec throughout. Compared with the previous architecture [7], its area decreases about 10% and critical path delay decreases about 20%. The architecture proposed in this paper can be applied to a hardware accelerator or dedicated design for the H.264 video codecvideo codec design of H.264.

REFERENCES

[1] ITU-T Recommendation H.263: Video coding for low bit-communication Version 1November 1995; Version 2 (H.263, January 1998; 3(H.263++), November 2000.

[2] ISO/IEC 14496-2: Coding of audio-visual objects. Part 2: Visual. ISO/IEC JTC1. MPEG-4 Visual version 1, April 1999; Amendment 1 (Version 2), February 2000.

[3] ITU-T Rec. H.264 / ISO/IEC 11496-10 "Advanced Video Coding", Final Committee Draft, Document JVTG050, Pattaya ,March 2003.

[4] Henrique S. Malvar, Antti Hallapuro, Marta Karczewicz, and Louis Kerofsky, Low-Complexity Transform and Quantization in H.264/AVC, *IEEE transactions on circuits and systems for video technology*, Vol. 13, No. 7, July 2003.

[5] A. Hallapuro, M. Karczewicz, H. Malvar. Low complexity transform and quantization, *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG*, Jan. 2002, Docs. JVT-B038 and JVT-B039.

[6] Shen Cheng-dong, Li Tie-jun, Li Si-kun, The Performance and Complexity Analysis of The H.264 Video Coding Standard, CCVRV'2003, Sep. 2003.

[7] Tu-Chih Wang, Yu-Wen Huang , Hung-chi Fang, and Liang-Gee Chen, "PARALLEL 4x4 2D T ransform and inverse transform architecture for MPEG-4 AVC/H. 264", ISCAS 2003, Bangkok, Thailand, May 2003.

[8] C.S. Wallace. A Suggestion for Fast Multiplier, *IEEE, Trans. Electric Computer*, 1964.

[9] Kaihong Sun, Design and Implementation of a Module Generator for Low Power Multipliers, Master's Thesis, Division of Electronics Systems, Depart. of Electrical Engineering, Linköping Univ., Sweden, Sept. 25, 2003.

[10] Gary W. Bewick, Fast multiplication: algorithms and implementation, The Department of Electrical Engineering and the Committee on Graduate Studies of Stanford Univ., Feb. 1994.

[11] A. Madisetri, A.N. Willson. A 100 MHz 2-D 8x8 DCT/IDCT processor for HDTV applications, *IEEE Transaction on Circuits and Systems for Video Technology*, vol.5, no.2, pp. 158-165, Apr., 1995.

[12] VeriSilicon SMIC 0.18μm 1.8V/3.3V I/O Cell Library Databook, Version: 2.0, Aug 1, 2003.