A CONFIGURABLE IP CORE FOR INVERSE QUANTIZED DISCRETE COSINE AND INTEGER TRANSFORMS WITH ARBITRARY ACCURACY

Chi-Chia Sun†, Student Member, IEEE, Ce Zhang and Juergen Goetze, Member, IEEE

Information Processing Lab, Dortmund University of Technology, Germany
†e-mail: chichia.sun@tu-dortmund.de

ABSTRACT

An extended version of low-complexity IP Core for image/video transformations based on the CORDIC architecture is presented. This IP core is able to perform quantized 8×8 IDCT and quantized 8×8/4×4 H.264-inverse integer transforms on a configurable architecture by using only shift and add operations. Furthermore, the number for CORDIC iterations and compensation steps can be adjusted, which enables a trade–off between video compression quality in PSNR and computational complexity.

Index Terms— DCT, ICT, Quantization, CORDIC, Reconfigurable, FPGA, MPEG–4, H.264

I. INTRODUCTION

As the demand for multimedia devices has been growing explosively through the decades, many challenges have been posed in a System-on-Chip (SoC) design, such as low-power, quality awareness and multi-standard integration. So far, the Discrete Cosine Transform (DCT) is the main component of many modern Image/Video compression standards and applications (e.g., JPEG and MPEG–4) [1], [2]. Recently, in the H.264 Standard which is the latest video coding standard based on block-oriented motion compensation, an integer transform with block sizes of 4×4 pixels were used. Later, an integer transform using block sizes of 8×8 has been added for the adding HD-Videos profiles.

In the past, orthogonal CORDIC rotation method has already been used to approximate the multiplications in DCT by using only shift–add operations [3]–[5]. T.Y. Sung etc. implemented a VLSI design of CORDIC DCI/IDCT with five CORDIC processors [6], somehow H. Jeong etc. has also presented a similar method with six CORDIC processors [7]. C.C. Sun etc. proposed a low–power and high-quality CORDIC based Loeffler DCT (CLDCT) with three CORDIC processors and requires only 38 add and 16 shift operations by ignoring unnecessary CORDIC iterations. Nevertheless, it still obtained a transformation quality as good as the original Loeffler DCT [8]. Later, an integration of 4-point/8-point integer transforms in this CLDCT was presented, such that a configurable architecture for **Discrete Cosine and Integer Transform (DCIT)** was obtained. It can perform the multiplierless 1–D 8-point DCT and the 1–D 4–point/8–point integer transforms for multi-standard Video CODECs [9]. Instead of realizing the Quantized DCT (QDCT) architecture with multiplications [10], it utilized the CORDIC compensation steps to approximate the quantization procedure iteratively. As for CLDCT in [8], the reduction of iteration number results in a simple CORDIC based Scaler (CORDIC-Scaler) by involving limited CORDIC compensation iterations. This CORDIC-Scaler has five pipeline stages and requires only 8 add and 10 shift operations to approximate two scaling operations. The combination of two 1–D DCITs, a row-column transposition memory and four CORDIC-Scalers forms a 2–D Quantized DCIT (QDCIT) which requires in total only 120 add, 76 shift 60 mux operations.

In this paper, we present an Inverse QDCIT (IQDCIT) with seven configurable modules to save more than half arithmetic units using shared configurable architecture. This CORDIC based IQDCIT can support arbitrary scaling values for quantization by updating the LUTs for different codecs, which are based on scaler dequantizer, such as JPEG, MPEG–4, H.264 or DVIX. After that, we apply different numbers of CORDIC iterations to the Inverse DCIT (IDCIT) and different numbers of compensation steps to the CORDIC-Scaler, respectively, because in most image/video decoders, the required transformation precision is dependent on the target resolution. For example, the concept of dynamic Bit-width adaptation by decreasing the bit length of the high frequency coefficients in DCT can also be adopted into our design [11]. Therefore, we can perform fewer iterations when the target resolution is small (QCIF/CIF). On the contrary, when the target resolution is large (Full-HD/Ultra-HD), we will apply more iterations. Providing arbitrary accuracy for various resolutions is important for embedding video decoders in any consumer electronic devices. The MPEG–4 and H.264 experimental results show that the proposed IQDCIT cannot only perform multiplierless 8×8 IQDCT and 4×4/8×8 inverse integer transforms but also contains configurable modules such that it can adjust the number of iterations for arbitrary accuracy. Furthermore, it still retains a good transformation quality compared to the default methods in terms of PSNR.

978-1-4244-7456-4/10/$26.00 ©2010 IEEE

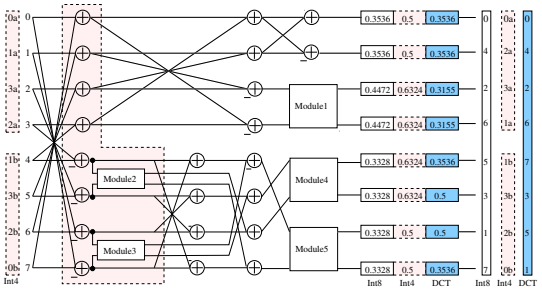


Fig. 1. Flow graph of an 8-point forward DCIT Transform with five configurable modules for multiplierless DCT and integer transforms [9].

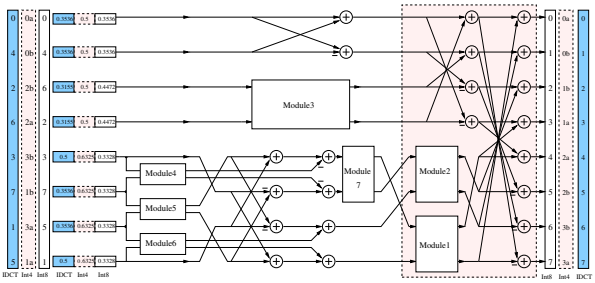


Fig. 2. Flow graph of an 8-point IDCIT Transform with seven configurable modules for IDCT and inverse integer transforms.

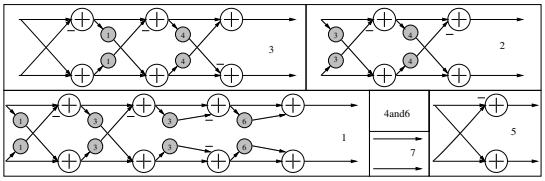
II. DCT AND INTEGER TRANSFORMS

II-A. Integration of DCT and Integer Transforms

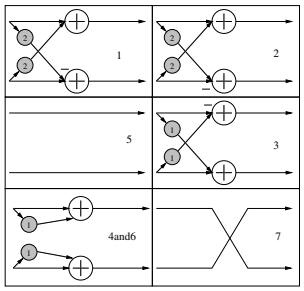
A combination of the 1-D 4-point/8-point integer transforms and the 1-D DCT transformation is presented based on a simplified 1-D 8-point CLDCT which requires only 38 add and 16 shift operations [8], where Fig. 1 shows the schematic view of the 1–D DCIT transform with configurable modules [9]. The five modules, which can execute different operations, are placed for performing both the multiplierless DCT and integer transforms. This DCIT can execute an 8-point DCT transform with a dedicated sub flow graph where the Module 1–3, 5 show the original operations for the CLDCT and Module 4 is a bypassing circuit. Next, we have simplified these five modules to perform one 8-point integer transform and two 4-point integer transforms by reusing the reconfigurable modules. The different order of input/output and scaling values are represented by the highlighted numbers and the emphasized boxes. Note that the order of input/output signals for 4-point integer transform are especially indexed by **a** and **b** for two different subsets.

II-B. Inverse DCIT

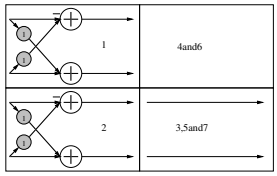
In contrast to the forward DCIT, we implemented an inverse version in a similar way as shown in Fig. 2. The flow diagram shows that the IDCIT can execute an 8-point IDCT, two 4-point inverse integer transforms and an 8-point inverse integer transform. The order of inputs and the corresponding coefficients for each input signal are marked at the beginning. There are seven configurable modules that perform various operations for different transform modes. For the IDCT mode, the Module 1–3, as shown in Fig. 3(a), are represented based on the CORDIC algorithm. The length of signal flows in these three modules depends on the iteration numbers and the amount of corresponding compensation steps. Module 4 and Module 6 are assigned by zeros directly. Module 5 is a butterfly operation for connecting with the Module 1–2. Module 7 is a bypassing circuit. In the mode of 8-point inverse integer transform as shown in Fig. 3(b), the Module 5 will bypass the input signals and then the



(a) Function of the modules for an 8-point inverse CLDCT.



(b) Function of the modules for an 8-point inverse integer transform.



(c) Function of the modules for two 4-point inverse integer transforms.

Fig. 3. Three sub flow graphs of the modules of Fig. 2.

Module 4 and the Module 6 will perform one step CORDIC compensation for the following add and sub operations. Then the signals are switched in Module 7. At the same time, Module 1–3 will perform one CORDIC iteration. Moreover, Fig. 3(c) shows the configuration of two 4-point inverse integer transforms. Note that only the area bordered by dashed lines in Fig. 2 will be activated, other parts are bypassed directly. Therefore, the input signals will be shifted to the highlighted area and further processed by Module 1-2.

III. THE 2–D INVERSE QDCIT

III-A. The 2–D Inverse Framework

Fig. 4 shows how the components of our proposed 2–D IDCIT transformation and the dequantization modules are connected to each other. In this framework, input pixels will first be dequantized by four CORDIC-Scalers. There is a CORDIC-Scaler configurator module, which configures the four parallel modules, to perform the scaling operations

dynamically. For each CORDIC-Scaler, two of the input signals are connected to the input of CORDIC-Scaler, which can scale up and dequantize the current input coefficients [9]. The configuration of each CORDIC-Scaler is dependent on the type of transformation to be executed, on the current scaling factor, and on the dequantization level. After that, a sub–block s indicated by dashed lines, which gives an overview of the 2–D IDCIT transform core. Inside the core, there is an 1-D IDCIT followed by a row-column transpose memory, then followed by another 1-D IDCIT to perform the 2-D multiplierless IDCT and inverse integer transforms. The first and second 1–D IDCIT transform is the transform core that we already described in Fig. 2.

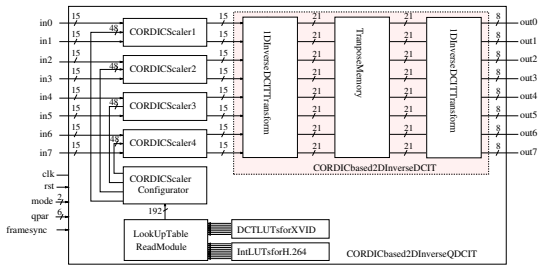


Fig. 4. Framework of a CORDIC based 2-D IQDCIT.

III-B. Variable Iteration Steps of CORDIC

Since the presented design is executed based on CORDIC algorithm iteratively, we can further adjust the iteration numbers of trigonometric operations in the IDCIT (module 1–3) and the compensation steps of CORDIC-Scaler. Table I shows the computational complexity if different numbers of CORDIC iterations are applied, where IQDCIT–S1 is a simplified version from our IQDCIT with only 2 iterations/steps and IQDCIT–S2 has 2–3 iterations/steps. The number of iterations/steps is treated as a variable influencing the precision of the transformation. On the other hand, IQDCIT–C1 extends the number to 6–8 iterations/steps in order to obtain a better quality. Note that the number of iteration is selected by the brute force search with a target resolution. Clearly, the default IQDCIT cannot only reduce the arithmetic units but also further integrates the dequantization function. In [5], [6], besides adders for butterfly operations, additional full CORDIC processors need to count into the total required number for the 2–D transformation (80 shift–add for each pipelined CORDIC). Eventually, these CORDIC DCTs require much more shift–add operations than the proposed IQDCIT.

IV. EXPERIMENTAL RESULTS

IV-A. RTL Synthesized Results

We have modeled a fully pipelined CORDIC based 2-D IQDCIT in FPGA. Table II shows the synthesis results. The critical path delay which came from the CORDIC-Scaler component is down to 5.099ns and results in a maximum frequency 196.1 MHz. There are some important points that can be observed. First, it occupies a very small size of the area in FPGA, because the CORDIC algorithm enables a simple architecture for implementation. Second, it can achieve very high throughput up to 1.57 Giga pixels. Finally, the FPGA implementation results show that the presented IQDCIT architecture can provide a good solution for integration of different transformations and their dequantization methods by utilizing the CORDIC algorithm to share the hardware resources as much as possible.

Table I. 2–D Transformation Complexity for arbitrary CORDIC iterations.

Type/Operation	Mul	Add	Shift	Mux
Novel IQDCT [10]	32	52	8	0
scaled CORDIC DCT [5]	4	64 + 80×6	80×6	0
CORDIC DCT/IDCT [6]	0	36 + 80×10	80×10	0
CORDIC based DCT [7]	0	208	176	0
CORDIC Loeffler DCT [8]	0	76	32	0
IQDCIT–S1	0	92	48	44
IQDCIT–S2	0	112	68	52
IQDCIT (default)	0	124	80	60
IQDCIT–C1	0	184	136	76

Table II. The logical utilization for each component of the 2-D IQDCIT (Xilinx XUPV5–XC5VLX110T).

Module	Slice	Regs	LUTs
2–D IDCIT	2,361	6,258	7,426
C-Scaler×4	2,728	2,069	6,485
Configur.	339	182	744
LUTs etc.	314	384	586
2–D IQDCIT	6,806	8,896	15,415
XC5VLX110T	69,120	69,120	17,280

IV-B. Performance in MPEG–4 XVID and H.264

We tested the proposed 2-D IQDCIT with the video coding standard MPEG–4 by using a publicly available XVID CODEC 1.2.2 software. The default IDCT in the XVID is based on Chen-Wang algorithm using integer multiplications and MEPG4 dequantization “method 2”. In contrast, we have first replaced the default IDCT and the dequantizer by our IQDCIT architecture and then simulated with some well-known video sequences to show the transformation quality in PSNR. We have also embedded it into the H.264 JM-16.1 reference software. We first encoded the video sequences with a default encoder of two software, then decoded them by using the default decoder and the IQDCIT with different iteration numbers and compensation steps from Table I.

Fig. 5 and Fig. 6 illustrate the average PSNR of the “foreman”, “paris” and “news” cif video test sequences from low to high bitrates in XVID and H.264. The video simulation results of the IQDCIT-S1 and IQDCIT-S2 are in

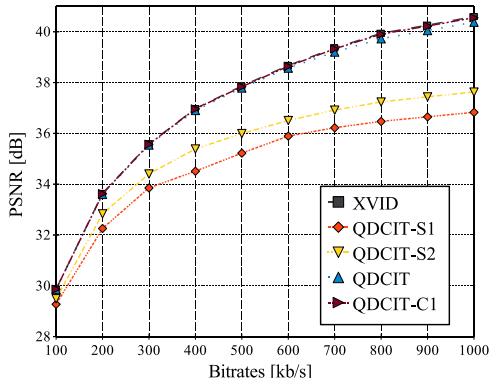


Fig. 5. The average PSNR from low to high bitrates in XVID.

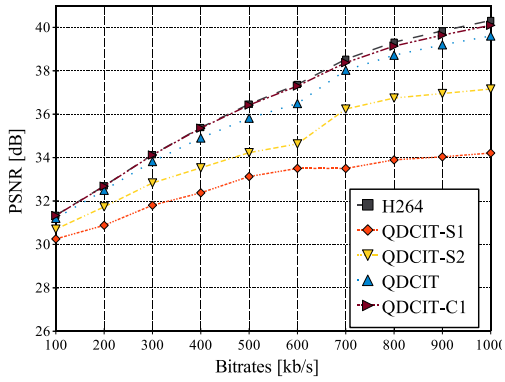


Fig. 6. The average PSNR from low to high bitrates in H.264.

average 3.17 dB and 1.97 dB lower than the default setting. Meanwhile, the IQDCIT is very similar but only drops off 0.31 dB and the average PSNR degradation with QDCIT-C1 is negligible. The IQDCIT-S1 and the IQDCIT-S2 are fitted small resolutions or low bitrates. However, when the bitrates are high, QDCIT and QDCIT-C1 are preferred. Both XVID and H.264 simulations show that the proposed IQDCIT architecture can provide a good compromise between PSNR and computational effort, and is also highly satisfactory compared to the reference implementations in terms of video quality.

V. CONCLUSION

In this paper, a low-complexity and highly-integrated IQDCIT based on the CORDIC algorithm with arbitrary iteration numbers is presented. The proposed 2–D IQDCIT architecture requires only 124 add, 80 shift and 60 mux operations to perform the multiplierless 8×8 IQDCT and quantized 4×4/8×8 inverse integer transform by sharing the hardware resources as much as possible. According to the implementation results, it requires much less computational efforts than other CORDIC DCTs and achieves a very high throughput. Moreover, the video simulation results show that the proposed architecture with arbitrary iteration steps can retain the transformation quality as good as the conventional designs dependent on the target resolution. Therefore, the proposed IP Core is very suitable for the low-complexity and multi-stander decoder in SoC designs.

VI. REFERENCES

[1] R. C. Conzalez and R. E. Woods, *Digial Image Processing*, Prentice-Hall, 2001.

[2] Iain E. G. Richardson, *H.264 and MPEG-4 Video Compression*, John Wiley & Sons, 2003.

[3] Jack E. Volder, "The CORDIC trigonometric computing technique," *IRE Transactions on Electronic Computers*, vol. EC-8, pp. 330–334, 1959.

[4] Feng Zhou and P. Kornerup, "High speed DCT/IDCT using a pipelined CORDIC algorithm," in *Symposium on Computer Arithmetic*, July 1995, pp. 180–187.

[5] Sungwook Yu and Jr. Swartzlander, E.E., "A scaled DCT architecture with the CORDIC algorithm," *Signal Processing, IEEE Transactions on*, vol. 50, no. 1, pp. 160 –167, Jan. 2002.

[6] Tze-Yun Sung, Yaw-Shih Shieh, Chun-Wang Yu, and Hsi-Chin Hsin, "High-Efficiency and Low-Power Architectures for 2-D DCT and IDCT Based on CORDIC Rotation," in *International Conference on Parallel and Distributed Computing, Applications and Technologies*, Dec. 2006, pp. 191–196.

[7] Hyeonuk Jeong, Jinsang Kim, and Won Kyung Cho, "Low-power multiplierless DCT architecture using image correlation," *IEEE Transactions on Consumer Electronics*, vol. 50, no. 1, pp. 262–267, Feb. 2004.

[8] Chi-Chia Sun, Sanq-Jang Ruan, Benjamin Heyne, and Juergen Goetze, "Low-power and high-quality Cordic-based Loeffler DCT for signal processing," *IET Circuit Devices & System*, vol. 1, no. 6, pp. 453–461, Dec. 2007.

[9] Chi-Chia Sun, Philipp Donner, and Jürgen Götze, "Low-complexity multi-purpose IP Core for quantized Discrete Cosine and integer transform," in *IEEE International Symposium on Circuits and Systems*, May 2009, pp. 3014–3017.

[10] Hanli Wang, Ming-Yan Chan, S. Kwong, and Chi-Wah Kok, "Novel quantized DCT for video encoder optimization," *IEEE Signal Processing Letters*, vol. 13, no. 4, pp. 205–208, Apr. 2006.

[11] Jongsun Park, Jung Hwan Choi, and Kaushik Roy, "Dynamic bit-width adaptation in DCT: image quality versus computation energy trade-off," in *Design, Automation and Test in Europe*, Mar. 2006, pp. 520–521.