RECONFIGURABLE DISCRETE COSINE TRANSFORM PROCESSOR FOR OBJECT-BASED VIDEO SIGNAL PROCESSING

Po-Chih Tseng, Chao-Tsung Haung, and Liang-Gee Chen

DSP/IC Design Lab, Graduate Institute of Electronics Engineering,
Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan
E-Mail:{pctseng, cthuang, lgchen}@video.ee.ntu.edu.tw

ABSTRACT

In this paper, a novel reconfigurable discrete cosine transform processor is proposed to perform the Shape-Adaptive DCT (SA-DCT) algorithm, which is a dominating computational kernel of various object-based video signal processing. The SA-DCT algorithm is analyzed in detail to capture important architecture design issues and then design efficient hardware architecture. A dynamically reconfigurable datapath is proposed to meet the run-time reconfigurable computational requirement of SA-DCT. Besides, a prototyping chip has been implemented to prove the feasibility of proposed architecture. This reconfigurable DCT processor supports complete functions of SA-DCT and therefore suitable for high-end and high-quality object-based video applications.

1. INTRODUCTION

Object-based video coding, such as MPEG-4 natural video coding [1], provides not only significantly better coding efficiency but also rich functionalities for object-based manipulation and interactivity. In order to support object-based video coding, several new coding techniques have been developed for the description of image regions, which are no longer squared as in conventional block-based video coding but are allowed to be of arbitrary shape [2]. Among these techniques, the Shape-Adaptive DCT (SA-DCT) proposed by Sikora [3] was reported to be able to achieve better coding efficiency and has also been adopted by MPEG-4. In addition to video coding, SA-DCT has also been applied to various object-based video signal processing applications, such as region-based fractal image compression [4] and object-based video watermarking [5].

Since SA-DCT has its efficiency advantages mainly at high bit-rates, it clearly aims at high quality video applications, such as Digital TV and HDTV. These high-end applications highly demand hardware acceleration solutions rather than pure software solutions. Therefore, a hardware accelerator of SA-DCT would be a necessity for object-based video systems. In the literature, there were numerous proposals for hardware realizations of conventional DCT, such as matrix decomposition based architecture [6] and distributed arithmetic based architecture [7]. However, these proposals targeted only at squared block-size DCT and unable to be extended to support SA-DCT. Although there was one proposal taregting SA-DCT, it did not equip with complete functions of SA-DCT but only part of them [8].

In this paper, we propose a reconfigurable DCT processor for object-based video signal processing, which is a complete hardware architecture of SA-DCT based on dynamically reconfigurable datapath. The algorithm analysis, architecture design, and chip implementation of proposed architecture will be discussed in detail in the following sections.

This work was supported in part by MOE Program for Promoting Academic Excellence of Universities under the grant number 89E-FA06-2-4-8, in part by National Science Council, Republic of China, under the grant number 91-2215-E-002-035, and in part by MediaTek Inc.

2. ALGORITHM ANALYSIS

SA-DCT is applied to those 8x8 blocks located on the boundary of an arbitrarily shaped image segment. If the segment is a squared 8x8 region, then SA-DCT is equivalent to conventional 8x8 DCT. The idea of SA-DCT is to apply 1-D DCT transforms vertically and then horizontally according to the number of active pixels in the rows and columns of the block, respectively.

Fig. 1 illustrates the processing flow of SA-DCT forward transform. Fig. 1(a) shows a possible example of an 8x8 image block segmented into two regions, foreground (black part) and background (light part). To perform the vertical transformation of the foreground, the length of each column of the foreground is calculated, and the columns are shifted and aligned to the upper border as shown in the gray region of Fig. 1(b). Then as in Fig. 1(c), depending on the number of active pixels N of each particular column, a 1-D N-point DCT is applied to the first N pixels of the column. After the vertical transformation, the lowest DCT coefficients (DC values) of columns are produced as in Fig.

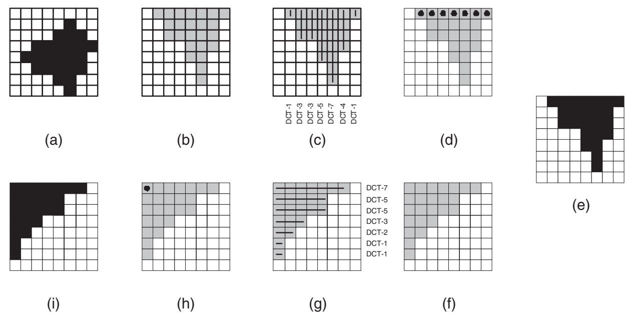

Fig. 1. Processing Flow of SA-DCT Forward Transform

1(d), and the segment distribution is shown in Fig. 1(e). To perform the horizontal transformation, the length of each row is calculated, and the rows are shifted and aligned to the left border as shown in Fig. 1(f). Then, horizontal 1-D N-point DCT adapted to the length of each row is performed as shown in Fig. 1(g). After the horizontal transformation, The lowest DCT coefficient (DC value) for this image segment is produced as in Fig. 1(h) and the final location of the resulting DCT coefficients is shown in Fig. 1(i). The SA-DCT inverse transform is similar to forward transform with reverse processing flow. Based on above description, SA-DCT algorithm can be divided into four steps from the computational point of view.

Step 1: Vertically shift and align all active pixels of each column to the most upper position.

Step 2: Applly variable-length 1-D DCT to each column of the vertically shifted and aligned texture data with length equal to the number of active pixels.

Step 3: Horizontally shift and align all active intermediate coefficients of each row to the most left position.

Step 4: Applly variable-length 1-D DCT to each row of the horizontally shifted and aligned texture data with length equal to the number of active pixels.

3. ARCHITECTURE DESIGN

According to the algorithm analysis in Sec. 2, several architecture design issues are addressed and summarized as follows:

A.Row-Column Method: Since both forward and inverse transforms of SA-DCT are performed by 1-D variable-length DCT/IDCT separately along one direction then the other, the suitable 2-D architecture design choice would be the row-column method. Direct 2-D method which induces more complex data flow would further complicate SA-DCT computation. Besides, the separability property of 2-D DCT/IDCT exploited by row-column method allows the transform to be applied along one dimension then the other. This property is identical to the inherent separable computational characteristic of SA-DCT.

B.Formation of Segment Distribution Statuses: In order to provide proper segment distribution statuses for both forward and inverse transforms of SA-DCT, the formation of segment distribution statuses according to the original shape information should be carefully considered.

C.Alignment of Input Texture or Coefficient Data: In order to efficiently perform the shift (re-shift) and align (realign) operations in forward (inverse) transform of SA-DCT, the alignment operation of input texture or coefficient data should also be taken into account.

D.Variable-Length 1-D DCT/IDCT: The variable-length 1-D DCT/IDCT are the kernel operations of SA-DCT. In order to efficiently perform these operations, specially-designed flexible hardware architecture must be considered.

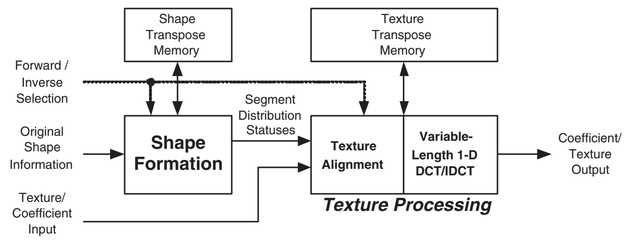
Shape Transpose Memory
Texture Transpose Memory
Forward / Inverse Selection
Original Shape Information
Texture/ Coefficient Input
Shape Formation
Segment Distribution Statuses
Texture Alignment
Variable-Length 1-D DCT/IDCT
Texture Processing
Coefficient/ Texture Output

Fig. 2. Architecture of Reconfigurable DCT Processor

Based on these design issues, we propose a reconfigurable DCT processor shown in Fig. 2 for SA-DCT. The Shape Formation receives the original shape information as input. Based on the original shape information and forward/inverse selection control signal, proper segment distribution statuses will be generated and sent to Texture Processing. The Shape Transpose Memory is used by Shape Formation to transpose the shape information from one direction to another. There are two modules within Texture Processing: the Texture Alignment and Variable-Length 1-D DCT/IDCT. When Texture Processing receives the texture or coefficient input data and segment distribution statuses generated by Shape Formation, the Texture Alignment will perform the alignment of texture or coefficient data. The shifted and aligned input data are then processed by the Variable-Length 1-D DCT/IDCT to perform corresponding DCT operations. The Texture Transpose Memory is used by the Texture Processing to transpose input data in row-column method. The more detailed operations of proposed architecture will be discussed in following subsections.

3.1. Shape Formation

Shape Formation generates necessary segment distribution statuses for texture data processing in row direction and column direction according to original input shape information. Fig. 3 shows the architecture of shape formation. Original shape information of processed 8x8 block ordered in column direction is taken as input. The input is first processed by Shape Counter, in which the number of pixels of each column within object is calculated. The calculated shape number is then used as input of Shift and Align PLA table to look up the shifted and aligned segment distribution value. The output then becomes column-wise shifted and aligned shape information ordered in column direction. The output of Shift and Align PLA is used as input and one-bit by one-bit written into the Shape Transpose Memory. At the same time, the transposed shape information is read out from the Shape Transpose Memory. This transposed shape information is the required column-wise shifted and aligned shape information ordered in row direction.Since the required successive order in forward and inverse transforms of SA-DCT is different, these two shape information are thus processed by the Delay Line to adjust the proper output timing phases. Forward Inverse Selection control signal is then used to select out the corresponding segment distribution status for the first and second directions in forward and inverse transform.

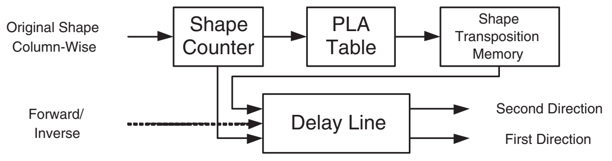

Fig. 3. Architecture of Shape Formation

3.2. Texture Processing

Texture Processing processes the input 2-D texture or coefficient data according to the segment distribution statuses to generate the 2-D transformed coefficient or reconstructed texture data. As shown in Fig. 4, There are three key modules in Texture Processing:

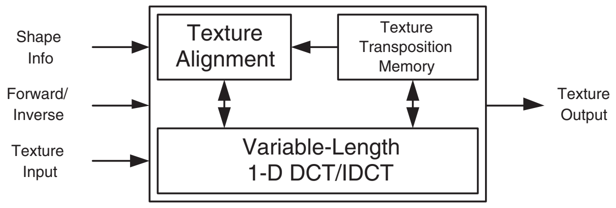

Fig. 4. Architecture of Texture Processing

A.Texture Alignment: According to the segment distribution statuses, the input 2-D texture or coefficient data can be (re-)shifted and (re-)aligned column by column or row by row to the corresponding shifted and aligned position.

B.Variable-Length 1-D DCT/IDCT: The Variable-Length 1-D DCT/IDCT processes the input 2-D texture or coefficient data column by column or row by row with corresponding length.

C.Texture Transpose Memory: Due to the row-column method approach for 2-D transform, the Texture Transpose Memory can transpose the processed 8x8 block data from column order to row order in forward transform or row order to column order in inverse transform.

The major challenge of architecture design for Texture Processing is to adaptively process the input texture or coefficient data according to segment distribution statuses generated by Shape Formation. During run-time processing, the datapath of Texture Processing has to be dynamically reconfigured as different hardware configurations according to different segment distribution statuses. In order to meet this special computational requirement, we propose a dynamically reconfigurable datapath for Texture Processing. Fig. 5 shows the basic concept of proposed dynamically reconfigurable datapath.

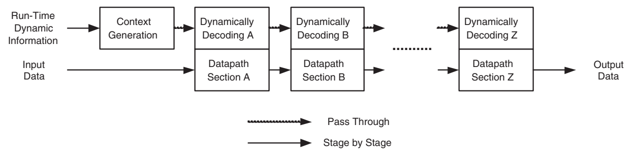

Fig. 5. Dynamically Reconfigurable Datapath

The basic concept of proposed dynamically reconfigurable datapath is to use the so-called "context" information to pass the run-time dynamic information throughout entire datapath. As shown in Fig. 5, the run-time dynamic information is processed by the Context Generation to generate the context information passed throughout entire datapath. Assume that the entire datapath is composed of several sections such as the Section A – Z in Fig. 5. The context information is then passed through these datapath sections. When the context information arrives at one datapath section, a corresponding dynamically decoding logic takes this context as input to decode the dynamic control signals to the datapath section for dynamically reconfiguration. At the same the, the dynamically reconfigured datapath section takes the entire datapath input or previous section output as the input to perform the data processing in current configuration. Based on the basic concept of proposed dynamically reconfigurable datapath, the detailed architecture of Texture Processing in forward transform configuration is illustrated in Fig. 6.

4. CHIP IMPLEMENTATION

A prototyping chip has been implemented to prove the feasibility of proposed architecture. This chip was implemented under TSMC 0.35μm CMOS 1P4M process by cell-based

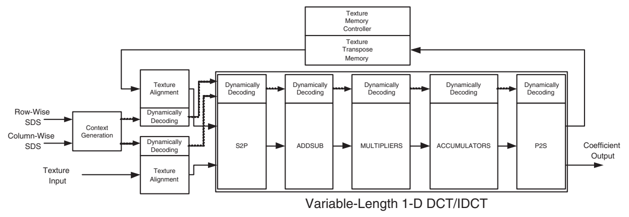

Fig. 6. Detailed Architecture of Texture Processing

IC design flow. Table 1 shows the key features of the prototyping chip and Fig. 7 shows the photograph. The chip could operate up to 66MHz to achieve 66M samples/sec throughput rate.

Table 1. Key Features of Prototyping Chip

Technology	TSMC 0.35μm 1P4M CMOS Process
Package	80 CQFP
Die Size	2.75 x 2.75 mm ^{2}
Core Size	1.93 x 1.93 mm ^{2}
Transistor Count	160K
Max Clock Rate	66MHz
Power Consumption	180mW @ 3.3V, 66MHz

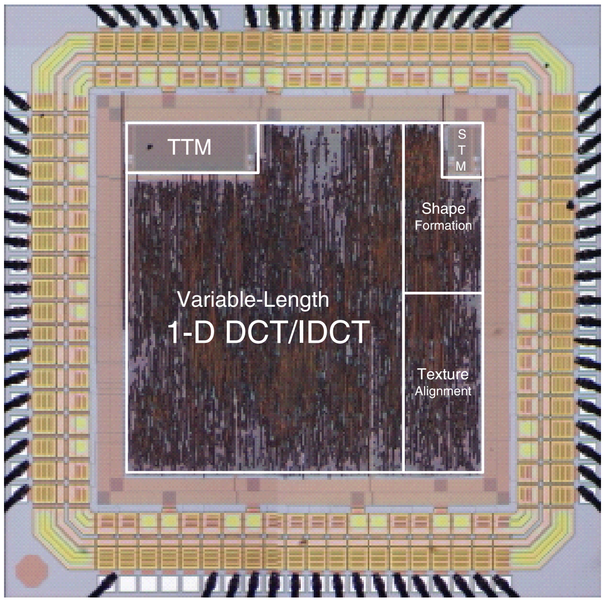

Fig. 7. Photograph of Prototyping Chip

5. CONCLUSION

We have proposed a reconfigurable DCT processor for SA-DCT. The SA-DCT algorithm is analyzed in detail to capture important architecture design issues and then design efficient hardware architecture. A dynamically reconfigurable datapath is proposed to meet the run-time reconfigurable computational requirement of SA-DCT. Besides, a prototyping chip has been implemented to prove the feasibility of proposed architecture. This reconfigurable DCT processor supports complete functions of SA-DCT and therefore suitable for high-end and high-quality object-based video applications.

6. REFERENCES

[1] *Information Technology – Coding of Audio-Visual Objects – Part 2: Visual, ISO/IEC 14496-2*, 2001.

[2] A. Kaup, "Object-based texture coding of moving video in mpeg-4," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 1, pp. 5–15, Feb. 1999.

[3] T. Sikora and B. Makai, "Shape-adaptive dct for generic coding of video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 1, pp. 59–62, Feb. 1995.

[4] K. Belloulata and J. Konrad, "Fractal image compression with region-based functionality," *IEEE Transactions on Image Processing*, vol. 11, no. 4, pp. 351–362, Apr. 2002.

[5] J. Guo and P. F. Shi, "Object-based watermarking scheme robust to object manipulations," *IEE Electronics Letters*, vol. 38, no. 25, pp. 1656–1657, Dec. 2002.

[6] A. Madisetti and Jr. A. N. Willson, "A 100mhz 2-d 8x8 dct/idct processor for hdtv applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 2, pp. 158–165, Apr. 1995.

[7] M. T. Sun, T. C. Chen, and A. M. Gottlieb, "Vlsi implementation of a 16x16 discrete cosine transform," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 4, pp. 610–617, Apr. 1989.

[8] T. Le and M. Glesner, "Flexible architectures for dct of variable-length targeting shape-adaptive transform," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 8, pp. 1489–1495, Dec. 2000.