A LOW MEMORY QCB-BASED DWT FOR JPEG2000 COPROCESSOR SUPPORTING LARGE TILE SIZE

Bing-Fei Wu and Chung-Fu Lin

Department of Electrical and Control Engineering
National Chiao Tung University
1001 Ta Hsueh Road, Hsinchu, Taiwan, 30050, R.O.C

ABSTRACT

JPEG 2000, which provides a higher compression ratio than the traditional JPEG, is an upcoming compression standard for still images. The experimental results imply that larger tile size used for JPEG 2000 results in better image quality. However, processing the large tile image requires more memory in the hardware implementation. To reduce the hardware resources, a QCB (quad code-block)-based DWT method is proposed to support the process of large tile image with low memory. Based on the QCB-based DWT engine, three code-blocks belonging to LH_0, HL_0 and HH_0 bands can be generated after each fixed time slice recursively and the EBC (embedded block coding) processors can directly process the three code-blocks. It can save the size of tile memory up to 75%. Moreover, the remaining 1/4 size of tile memory can be decreased through the zero holding extension for the unavailable data. That is, it only requires 24K bytes memory to support the process of 512x512 tile image, with slight image degradation, especially at low bit rates. The low memory requirement makes the hardware implementation practicable.

1. INTRODUCTION

JPEG 2000 provides higher compression ratio and more functions than traditional JPEG. It takes various functions (i.e. lossless, lossy, resolution, quality, ROI etc.) into a single coding stream. In general, the main coding stream has to be performed by DWT, CF (context formation), and MQ blocks, which can be regarded as the core algorithms of JPEG 2000 standard [1]. After getting the main compressed data, the rate-distortion optimization is applied to decide the optimal truncation points of the main coding stream. In general, using the large tile size parameter to perform JPEG 2000 compression gains higher compression ratio than using the small tile size parameter. However, processing the larger tile image also requires more memory in the hardware implementation.
Considering these three core blocks, the DWT process requires an entire tile memory to carry out the subband transformation [2]. Afterward, the CF process divides each subband into several code-blocks and performs the bit-plane coding [3,4,5]. The MQ coder then compresses the context-based information in a lossless way. Many studies have devoted to optimize the individual components. However, the overall encoding system may suffer performance degradation since different components have different I/O bandwidths and buffers [6,7]. Besides, the overall system requires more memory to process the large tile image. These bottlenecks are mainly caused by the difference coding flow between the DWT and EBC processes. In this paper, we propose a QCB-based DWT method to support the large tile size mode with low memory requirement. That is, for processing a 512x512 tile image, the size of internal tile memory can be reduced from 256K words to 12K words, with slight image degradation. Compared to the software results with 512x512 tile size [8], the proposed method can preserve the original image quality, especially at the low bit rates. Moreover, by changing the output timing of traditional DWT, three code-blocks are iteratively generated after each fixed time slot. In this way, the DWT and EBC processes can achieve higher parallelism.
The paper is organized as follows. Section 2 describes the brief concept of the core blocks of JPEG 2000. In Section 3, a detailed analysis of QCB-based DWT method is presented and the proposed architecture will be addressed. The simulation results are shown in Section 4. In Section 5, we compare the proposed architecture with other related works. A brief summary is given in Section 6.

2. JPEG 2000 BASIC BLOCKS

The basic coding flow of JPEG 2000 can be described as in Fig.1. During the encoding process, an image is split into several rectangular tiles and each tile can be coded independently. The 2-D DWT then decomposes a tile into LL_1, LH_0, HL_0, and HH_0 subbands. The LL_1 subband can be decomposed into next resolution, recursively. After the wavelet transform, the coefficients in each subband are partitioned into several code-blocks and processed by EBC independently. The EBC process carries out the CF and MQ algorithms. The CF algorithm codes the code-block bit-plane by bit-plane, and generates context-based information. Then, the context data are lossless coded by MQ coder to generate the main coding stream. After getting all the main compressed data, the rate-distortion optimization is applied to decide the optimal truncation points for the lossy compression.

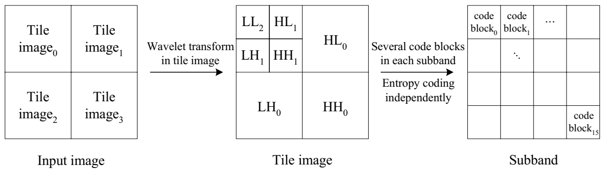

Fig.1 Block diagram of JPEG 2000 encoder.

In general, using the large tile size parameter to perform JPEG 2000 compression achieves better image quality than choosing the small tile size mode. Fig.2 shows the PSNR with different tile sizes. The image quality of 128x128 tile size could be inconspicuous when the compression ratio exceeds 50. Compared to the small tile image, the large tile image provides more possible truncation points for rate-distortion optimization and has less tile block effects. Thus, it can provide better image quality even at higher compression ratio. As shown in Fig.2, the 256x256 tile size exceeds 7 dB more than 128x128 tile size while compression ratio is 100. Based on the outperformance for processing the large tile image, it is a highly demand to design the hardware architecture to support the large tile size parameter.

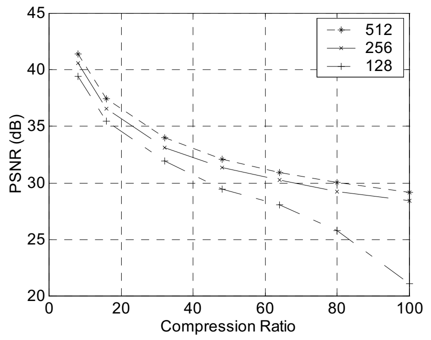

Fig.2 PSNR of different tile sizes of JPEG 2000.
(Lenna: 512x512 size, 4 level DWT decompositions of 5/3 filter)

The typical hardware implementation of JPEG 2000 coprocessor is shown in Fig.3 [6,7]. It requires an entire tile memory to carry out the 2-D DWT process and to provide the input coefficients to code-block memories for embedded block coding. To increase the throughput of bit-plane coding, multiple processors are used to execute the code-blocks in parallelism. Several EBC architectures are also proposed to realize the high computational bit-plane coding algorithm. These speed-up methods can be classified into three main ways: sample skip method, pass parallel, and bit-plane parallel. As shown in TABLE I, the pass parallel architecture is suitable for the system integration in terms of hardware cost and flexibility of data access.

Although each component is optimized individually, the overall system may still require large internal memory and suffer performance degradation while performing the large tile image. For example, to encode the 512x512 tile size image, 256K words (512x512x16 bits) memory is required to buffer the entire tile data, which makes the on-chip memory impracticable. Moreover, the parallelism between DWT and EBC is degraded since the DWT process requires more execution time to carry out the first level decomposition for the large tile image. That is, the latency for generating the coefficients of the LH_0, HL_0 and HH_0 bands becomes longer. Hence, the internal memory requirement and long latency between the DWT and EBC are the main bottlenecks of JPEG 2000 coprocessor to support the large tile size parameter.

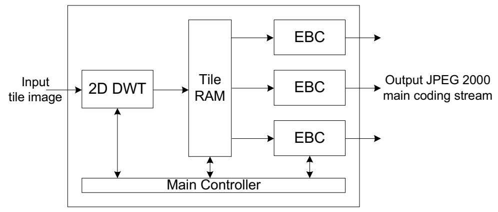

Fig.3 Typical hardware implementation of JPEG 2000 coprocessor.

TABLE I. The architectural model of different methods for EBC.
(L: code-block width, n: number of coding but plane, δ : 0 to 1)

Architecture	Average processing time	Number of processors	
		CF	MQ
Sample skip method [3]	$1.3 \times n \times L^2$	1	1
Pass parallel [4]	$n \times L^2$	1	1
Bit-plane parallel [5]	$(1+\delta) \times L^2$	10	5

3. PROPOSED QCB-DWT ARCHITECTURE

To reduce the size of internal tile memory, we propose the QCB (quad code-block)-based DWT approach. The basic idea of the QCB- based DWT method is to produce the complete code-block data as soon as possible such that EBC processors can directly carry out the bit-plane coding to decrease the internal memory size.

Based on this idea, a tile image is divided into several QCB blocks in advance of DWT procedure, as shown in Fig.4. The QCB block$_0$ carries out the QCB-DWT process and generates four code-blocks --- three for EBC, and one for next DWT decomposition, recursively. Thus, three EBC processors can individually process the three code-blocks belonging to the different subbands at the same time and the size of internal tile memory can be reduced by a factor of 4. However, since the tile image is divided into several QCB blocks, the original data path of DWT is broken at the boundary of two neighboring QCB blocks. To recover the primitive data path, it can be solved by processing some of previous data [9]. For the case of 5/3 filter, the original data path for the QCB block can be restored by performing two past data. To further reduce the size of internal memory storing the LL_1 band data, a simple prediction method is applied to predict the data belonging to the neighbor QCB block. Fig. 5 shows the data flow of the LL_1 band. Once the QCB $block_{16}$ of LL_1 band is obtained, it can be decomposed to the next DWT resolution immediately and produce three complete code-blocks. Since the QCB blocks near the QCB $block_{16}$ are not available, we use the zero holding extension to predict the unavailable data based on the continuous property of image. Fig.6(a) shows the periodic symmetric extension defined in JPEG 2000 standard to extend the signal and the method is used to the start and the end of each data path. To predict the unavailable data, we use the zero holding extension method, as shown in Fig. 6(b). Based on the prediction, a part of coefficients in LL_1 band can be decomposed once the QCB $block_{16}$ is completely obtained. The size of tile memory is proportional to the resolution value of DWT, i.e. only two QCB-size memories are required in this case – one is for LL_1 band, and the other one is for LL_2 band.

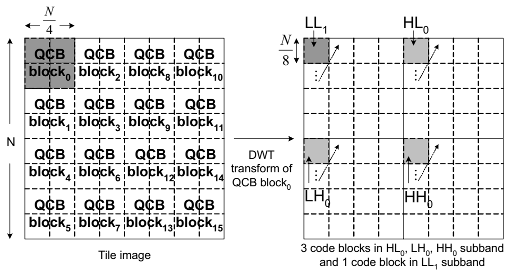

Fig.4 The QCB-DWT process in a tile image. (tile size : NxN, code-block size: N/8 x N/8, QCB size: N/4 x N/4)

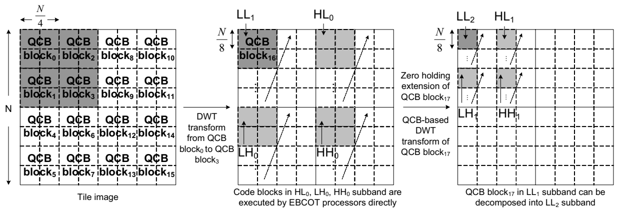
N/4
N
Tile image
DWT transform from QCB block0 to QCB block3
LL1
HL0
LH0
HH0
QCB block16
Code blocks in HL0, LH0, HH0 subband are executed by EBCOT processors directly
Zero holding extension of QCB block17
QCB-based DWT transform of QCB block17
LL2
HL1
LH1
HH1
QCB block17 in LL1 subband can be decomposed into LL2 subband

Fig.5 QCB-DWT for LL_1 band with zero holding extension.

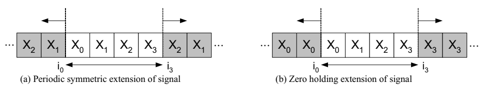
X2 X1 X0 X1 X2 X3 X2 X1
i0
i3
(a) Periodic symmetric extension of signal
X0 X0 X0 X1 X2 X3 X3 X3
(b) Zero holding extension of signal

Fig. 6 Periodic symmetric and zero holding extension of signal.

Fig. 7 shows the overall architecture. The tile memory is composed by several QCB-size memories. To support the 512x512 tile size with four level DWT decompositions, it requires three memory units (i.e. MEM_{LL1}, MEM_{LL2} and MEM_{LL3}) with 4K words to store the data of three QCB blocks belonged to the LL_1, LL_2 and LL_3 bands. Fig.8 presents the data flow of QCB-based DWT. Once MEM_{LL1}, MEM_{LL2} and MEM_{LL3} are selected, the zero holding extension is used to predict the unavailable data of the neighboring QCB blocks. TABLE II shows the internal memory size of the proposed architecture.

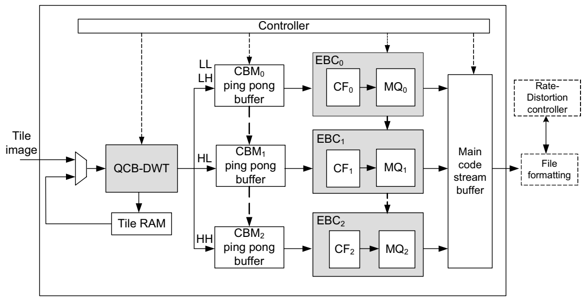

Fig.7 Architecture of QCB-based JPEG 2000 coprocessor.

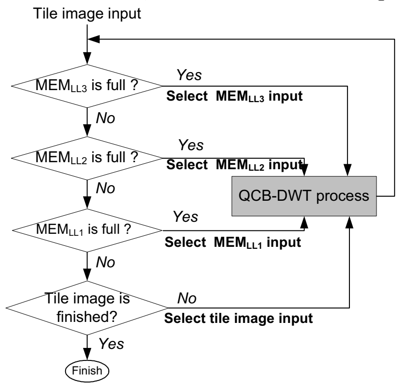

Fig.8 Flowchart of QCB-based DWT for JPEG 2000 coprocessor.

TABLE II. Memory requirement of the proposed architecture.

Assume: tile size= 512x512, code-block size= 32x32	
Memory type	Memory requirement (K bytes)
Tile memory (QCB-DWT)	3x64x64x16 (bits) = 24
Code-block Memory (CBM)	2x3x32x32x16 (bits) = 12

4. SIMULATION RESULTS

To simulate the image quality of QCB-based DWT, we use 512x512 size images performing the 5/3 filter with four level DWT decompositions. As shown in Fig.9, the PSNR of QCB-based DWT approaches to the traditional DWT used in JPEG 2000, especially in the low bit rates. Moreover, to support lossless or high bit rate compression, one can use the 128x128 tile size mode, since only 4K words memory is required to store the LL_1 band while the zero holding prediction would not be applied.

We also use a software version to evaluate the overall performance of JPEG 2000 coprocessor. Several 512x512 size images are chosen to assess the performance of the proposed architecture performing 4-level DWT decomposition (i.e. code-block size is 32x32). The throughput of DWT is dominated by the number of memory access. Due to the dyadic property of DWT, the number of memory access is decreased by a factor of 4 as shown in Fig.10(a). For the QCB-based DWT method, each QCB block requires 67x67x2 cycles to access the memory. The throughput of EC is based on the pass-parallel architectural model and the processing time is determined by the number of coding bit-plane [4]. As shown in Fig.10, the traditional DWT method requires $2N^2$ cycles to carry out the first DWT decomposition and only $0.5N^2$ cycles are overlapped with embedded block coding. Based on the QCB-based DWT approach, three EBC processors can execute the bit-plane coding after the first QCB block is processed. Since the maximal coding bit-plane is not larger than eight, each code-block can be executed within the processing time of the QCB block. TABLE III shows the execution cycles of several testing images. By changing the output timing of DWT, the proposed architecture has the higher parallelism between DWT and EBC processes.

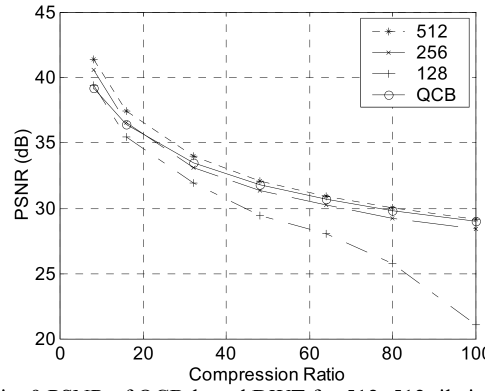

Fig. 9 PSNR of QCB-based DWT for 512x512 tile image.

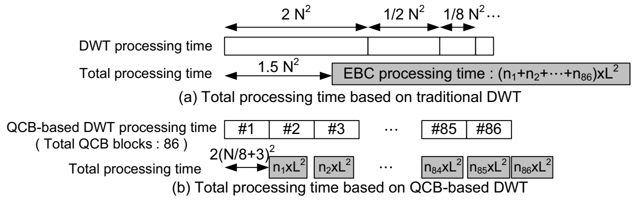
2 N²
1/2 N²
1/8 N²…
DWT processing time
Total processing time
1.5 N²
EBC processing time : (n₁+n₂+…+n₈₆)xL²
(a) Total processing time based on traditional DWT
QCB-based DWT processing time
(Total QCB blocks : 86)
#1
#2
#3
…
#85
#86
2(N/8+3)²
Total processing time
n₁xL²
n₂xL²
n₈₄xL²
n₈₅xL²
n₈₆xL²
(b) Total processing time based on QCB-based DWT

Fig. 10 Timing diagram of traditional DWT and QCB-based DWT. (n_i: the number of coding bit plane in the i-th code-block, N= 512, L= 32)

TABLE III. Performance of the QCB-based JPEG 2000 coprocessor.

Testing image	Overall performance (clock cycle)		Bit plane number		Reduced cycles
	QCB-based DWT	Traditional	Maximal	Average	
Lenna512	778490	888832	8	5.63	12.4 %
Baboon512	778490	997376	8	6.86	21.9 %
Pepper512	779514	926720	8	6.06	15.9 %
Airplane512	778490	909312	8	5.86	14.3 %

5. COMPARISONS

TABLE IV shows several architectures for JPEG 2000 coprocessor. To implement the JPEG 2000 algorithm with 128x128 tile size, [6,7] use the internal memory of 16K words to buffer the entire tile data. If the larger tile size is chosen, it requires additional external memory and the number of external memory access also increases.

By changing the output timing of traditional DWT, the proposed architecture can decrease the size of tile memory by a factor of 4. Moreover, to support the larger tile size, the zero holding extension is applied to decrease the remaining tile memory. That is, it can process the 512x512 tile image with four level decompositions by using the internal tile memory of 12K words. With the slight degradation of image quality, the QCB-based DWT still preserves the high quality of 512x512 tile size superior to 128x128 tile size, especial at low bit rates. The low memory requirement makes the implementation of on-chip memory practicable. Finally, the high parallelism between DWT and EC decreases the number of internal memory access between the tile and code-block memory.

TABLE IV. Comparisons of the different architectures.

Tile size	128x128		512x512	
Architecture	AMPHION [7]	ANDRA et.al. [6]	Proposed architecture	Proposed architecture
Tile memory	32	32	8	24
Code-block memory	12	6	12	12
Entropy Coder pairs	3	3	3	3

6. CONCLUSIONS

In this paper, we propose the QCB-based DWT to decrease the internal memory size. Based on the proposed method, three complete code-blocks can be produced after each fixed time slice, recursively. Thus, EBC processors can directly execute the bit-plane coding which saves 75% of the tile memory, while the remaining tile memory can be decreased through the zero holding prediction. With slight image degradation, the proposed architecture can support 512x512 tile size by using only 24K byte of tile memory, which makes the implementation of on-chip memory practicable.

7. REFERENCES

[1] ISO/IEC. ISO/IEC 15444-1. Information technology – JPEG 2000 image coding system, 2000.
[2] K. Andra, C. Chakrabati, and T. Acharya, "A VLSI Architecture for Lifting-Based Forward and Inverse Wavelet Transform," *IEEE Trans. on Signal Processing*, vol.50, no.4, pp. 966-977, April 2002.
[3] C.J. Lian, K.F. Chen, H.H. Chen, and L.G. Chen, "Analysis and Architecture Design of Block-Coding Engine for EBCOT in JPEG2000, " *IEEE Trans. on Circuits and Systems for Video Technology*., vol.13, no.3, pp. 219-230, March 2003.
[4] J.S. Chiang, Y.S Lin, and C.Y. Hsieh, "Efficient Pass-Parallel Architecture For EBCOT in JPEG2000, " in *Proc. IEEE International Symposium on Circuits and Systems*, vol. 1, pp. 773-776, May 2002.
[5] H.C. Fang, T.C Wang, C.J. Lian, T.H. Chang, and L.G. Chen, "High Speed Memory Efficient EBCOT Architecture for JPEG2000," in *Proc. IEEE International Symposium on Circuits and Systems*, Vol. 2, Thailand, pp. 736-739, May 2003.
[6] K. Andra, C. Chakrabati, and T. Acharya, "A High-Performance JPEG2000 Architecture," *IEEE Trans. on Circuits and Systems for Video Technol.*, vol.13, no.3, pp. 209-218, March 2003.
[7]AMPHION Products ---CS6510 JPEG2000 Encoder [Online]. Available: http://www.amphion.com/cs6510.html
[8] http://www.ece.uvic.ca/~mdadams/jasper/
[9] B.F. Wu and C.F. Lin, "Analysis and architecture design for high performance JPEG2000 coprocessor," in *Proc. IEEE International Symposium on Circuits and Systems*, vol. 2, pp. 225-228, May 2004.