Design and Analysis of Pipelined Discrete Wavelet Transform Architectures

宋志雲 謝曜式 林國珍 邱垂祿
Tze-Yun Sung Yaw-Shih Shieh Kuo-Jen Lin Cheui-Lu Chiu
Department of Microelectronics Engineering
Chung Hua University
Hsinchu, Taiwan 300-12
{bobsung,ysdaniel,kuojenlin}@chu.edu.tw

Abstract

This paper investigates the trade-offs between area, power and throughput (clock cycles) of several implementations of the discrete wavelet transform (DWT) using direct form in various sampling rates and pipelined architectures. The results of four different architectures synthesized, simulated and emulated on FPGA (Xilinx-XC2V6000). It is shown that the pipelined architectures provide the best area, power consumption, and throughput trade-offs under sampling rate, hardware utilization, and hardware.

These high-efficiency architectures are comprised of a transform module, an address sequencer, and a RAM module. The transform modules have uniform and regular structure, simple control flow, and local communication. According to the architecture with 2-samples per clock cycle, the power consumption of the architectures with 4- and 8-samples per clock cycle reduce power by 33%, but the hardware requirements are increased by 33%, 167% and 400%, respectively. The throughputs of the architectures with 4-, 8- and 16-samples per clock cycle are improved by 100%, 300%, 700%, respectively. These four proposed architectures are very suitable for VLSI implementation of new-generation image compression systems, such as JPEG-2000.

***Keywords*:** DWT direct cascading form, VLSI pipelined architecture, design trade-off, JPEG-2000.

1. INTRODUCTION

In the field of digital image processing, the JPEG-2000 standard uses the scalar wavelet transform for image compression [1]; and thus, the two-dimensional (2-D) discrete wavelet transform (DWT) has recently been used as a powerful tool for image compression systems. Since two-dimensional DWT requires massive computations, it requires a pipelined architecture to perform real-time or on-line video and image coding and decoding, as well as for implementation of high-efficiency application- specific integrated circuits (ASIC) or field programmable gate array (FPGA). At the heart of the compression (analysis) stage of the system is the DWT.

Cohen, Daubechies and Feauveau proposed using the biorthogonal 9/7 wavelet for lossy compression [1]. The symmetry of the biorthogonal 9/7 filters and the fact that they are almost orthogonal [2] make them good candidates for image compression application. Because the coefficients of the filter are quantized before hardware implementation, the multiplier can be replaced by limited quantity of shift registers and adders. Thus, the system hardware is saved, and the system throughput is improved significantly.

In this paper, we propose four high-efficiency architectures with 2-, 4-, 8- and 16-samples per clock cycle for the even and odd parts of 1-D decimated convolution. The advantages of the proposed architectures are 100% hardware-utilization, elimination of multipliers, regular structure, simple control flow and high scalability.

The remainder of this paper is organized as follows. Section 2 presents the 2-D discrete wavelet transform algorithm, and derives new mathematical formulas. In Section 3, four high-efficiency 2-D DWT architectures with different sampling rates and processing elements are proposed. Section 4 applies the four proposed 2-D DWT architectures to the coefficient quantization scheme for FPGA and VLSI implementation, analyzes their performance, and compares the performance of the proposed architectures and of previous works. Section 5 presents the design trade-offs of area, power consumption and throughput under the different sampling rates. Finally, the conclusions are given in Section 6.

2. 2-D DWT ALGORITHM

The 2-D DWT is a multilevel decomposition technique, and its mathematical formulas are defined as follows:

$$A^j(m,n)=\sum_{i=0}^{K-1}\sum_{k=0}^{K-1} l(i)\cdot l(k)\cdot A^{j-1}(2m-i,2n-k) \quad (1)$$

$$B^j(m,n)=\sum_{i=0}^{K-1}\sum_{k=0}^{K-1} l(i)\cdot h(k)\cdot A^{j-1}(2m-i,2n-k) \quad (2)$$

$$C^j(m,n)=\sum_{i=0}^{K-1}\sum_{k=0}^{K-1} h(i)\cdot l(i)\cdot A^{j-1}(2m-i,2n-k) \quad (3)$$

$$D^j(m,n)=\sum_{i=0}^{K-1}\sum_{k=0}^{K-1} h(i)\cdot h(k)\cdot A^{j-1}(2m-i,2n-k) \quad (4)$$

where $0\le n,\ m<N_j$; $A^0(m,n)$ is the input image; K denotes the length of filter; $l(i)$ denotes the impulse responses of the low-pass filter $G(z)$; $h(k)$ denotes the impulse responses of the high-pass filter $H(z)$, which are developed from $(K\times K)$ -tap filters; and $A^j(m,n)$, $B^j(m,n)$, $C^j(m,n)$, and $D^j(m,n)$ denote respectively the coefficients of low-low, low-high, high-low and high-high subbands produced at the decomposition level j (also represented by A^j , B^j , C^j , and D^j). In addition, $N_j\times N_j$ denotes samples of A^j .

According to the mathematical formulas (1), (2), (3) and (4), the decomposition is produced by four 2-D convolutions followed by the decimation both in the row and in the column dimension for each level. The data set A^{j-1} having $N_{j-1}\times N_{j-1}$ samples is decomposed into four subbands A^j, B^j, C^j, and D^j each having $N_j\times N_j$ (equals to $(N_{j-1}/2)\times(N_{j-1}/2)$) samples.

Let $a_m^j(2n)$, $l(i)l(2k)$, $l(i)h(2k)$, $h(i)l(2k)$ and $h(i)h(2k)$ be 1-D DWT consisting of the even-numbered samples. In addition, let $0\le n<N_j$; and $0\le k<K/2$. Moreover, let $a_m^j(2n+1)$, $l(i)l(2k+1)$, $l(i)h(2k+1)$, $h(i)l(2k+1)$ and $h(i)h(2k+1)$ be 1-D DWT consisting of the odd-numbered samples, and $0\le n<N_j$; $0\le k<K/2$. $a_{m,i}^j(n)$, $b_{m,i}^j(n)$, $c_{m,i}^j(n)$, and $d_{m,i}^j(n)$ can be expressed as follows:

$$a_{m,i}^j(n)=\sum_{k=0}^{\lceil K/2\rceil-1} l(i)l(2k)\cdot a_{2m-i}^{j-1}(2n-2k) + \sum_{k=0}^{K-\lceil K/2\rceil-1} l(i)l(2k+1)\cdot a_{2m-i}^{j-1}(2n-2k-1) \quad (5)$$

$$b_{m,i}^j(n)=\sum_{k=0}^{\lceil K/2\rceil-1} l(i)h(2k)\cdot a_{2m-i}^{j-1}(2n-2k) + \sum_{k=0}^{K-\lceil K/2\rceil-1} l(i)h(2k+1)\cdot a_{2m-i}^{j-1}(2n-2k-1) \quad (6)$$

$$c_{m,i}^j(n)=\sum_{k=0}^{\lceil K/2\rceil-1} h(i)l(2k)\cdot a_{2m-i}^{j-1}(2n-2k) + \sum_{k=0}^{K-\lceil K/2\rceil-1} h(i)l(2k+1)\cdot a_{2m-i}^{j-1}(2n-2k-1) \quad (7)$$

$$d_{m,i}^j(n)=\sum_{k=0}^{\lceil K/2\rceil-1} h(i)h(2k)\cdot a_{2m-i}^{j-1}(2n-2k) + \sum_{k=0}^{K-\lceil K/2\rceil-1} h(i)h(2k+1)\cdot a_{2m-i}^{j-1}(2n-2k-1) \quad (8)$$

The above equations imply that $a_{m,i}^j(n)$, $b_{m,i}^j(n)$, $c_{m,i}^j(n)$ and $d_{m,i}^j(n)$ can be computed as the sum of two 1-D convolutions performed independently on

the even part $a_{2m-i}^{j-1}(2n-2k)$ and the odd part $a_{2m-i}^{j-1}(2n-2k-1)$.

3. THE PROPOSED 2-D DWT ARCHITECTURES

The proposed architecture performs parallel and pipelined processing. Each compression level involves two stages: Stage 1 performs row filtering, and Stage 2 performs column filtering. At the first level, the size of the input image is $N \times N$, and the size of the output of each of the three subbands *LH*, *HL* and *HH* is $(N/2)\times(N/2)$. At the second level, the input is the *LL* subband whose size is $(N/2)\times(N/2)$, and the size of the output of each of the three subbands *LLLH*, *LLHL* and *LLHH* is $(N/4)\times(N/4)$. At the third level, the input is the *LLLL* subband whose size is $(N/4)\times(N/4)$, and the size of the output of each of the four subbands *LLLLLL*, *LLLLLH*, *LLLLHL* and *LLLLHH* is $(N/8)\times(N/8)$.

The coefficients of the low-pass filter and the high-pass filter have been derived for the biorthogonal 9/7 wavelet [3], and they are quantized before hardware implementation. We assume that the low-pass filter has four tapes: a_0, a_1, a_2 and a_3, and the high-pass filter also has four tapes: b_0, b_1, b_2 and b_3. The decimation filter for 1-D DWT is shown in Figure 1. According to eqs. (5), (6), (7) and (8), each 1-D decimated convolution can be computed as the point-wise sum of two 1-D convolutions performed independently. Figure 2 illustrates the transform module with 2-sample per clock cycle for 2-D DWT, the splitter arranges the data of the even and odd parts using processing element (PE). *f* represents the input frequency, *f*/2 denotes that the output frequency of L and H is a half of the input frequency**,** and *f*/4 denotes that the output frequency of LL, LH, HL and HH is a quarter of the input frequency. The single transform module can perform 2-D DWT. Figure 3 illustrates the 2-D DWT processor, which is comprised of a $(N/2\times N/2)$ RAM, a transform module, a multiplex, a splitter and an address sequencer. 42 clock cycles are required to perform the 2-D DWT. Clock cycles 0 to 31 perform the level-1 compression, clock cycles 32 to 39 perform the level-2 compression, and clock cycles 39 to 41perform the level-3 compression.

Figure 4 illustrates the transform module with 4-sample per clock cycle for 2-D DWT. Clock cycles 0 to 15 perform the level-1 compression, clock cycles 16 to 19 perform the level-2 compression, and clock cycles 20 to 21 perform the level-3 compression. Similarly, Figure 5 illustrates the transform module with 8-sample per clock cycle for 2-D DWT. It requires 11 clock cycles to perform the 2-D DWT. Clock cycles 0 to 8 perform the level-1 compression, clock cycles 9 to 10 perform the level-2 compression, and clock cycles 11 to 12 perform the level-3 compression. Table 1 shows that 6 clock cycles are required to perform the 2-D DWT processor with 16-sample per clock cycle. Clock cycles 0 to 3 perform the level-1 compression, clock cycle 4 performs the level-2 compression, and clock cycle 5 performs the level-3 compression.

Because of space limitation, we show a data flow of the architecture with 16-sample per clock cycle only.

4. HARDWARE IMPLEMENTATION AND PERFORMANCE ANALYSIS OF 2-D DWT

Filter coefficients are quantized before implementation in the high-speed computation hardware. The quantized biorthogonal 9/7 wavelet low-pass filter coefficients are used [4]. Values of the coefficients are shown in both decimal format and binary Booth recoded format (BBRF). All multiplications are performed using shifts and additions after approximating the coefficients as a BBRF. The multiplier is replaced by a

carry-save-adder (CSA) and three hardwire shifters in processing element (PE).

The hardware codes were written in Verilog®-hardware description Language (HDL) [5] running on SUN Blade 1000 workstation under ModelSim® simulation tool [6]. The architectures were synthesized by Xilinx® FPGA express tools [7] and evaluated on the Xilinx® XC2V6000 FPGA platform [8]. They were designed to evaluate the hardware and to provide an embedded core for digital image data compression [9].

The decimation filter for 1-D DWT requires seven adders, twelve shifters and three registers for each PE. This hardware is very cost-effective. The architectures with multiplierless reduce power dissipation by *m* compared with conventional architectures in *m*-bit operand (low-power utilization).

The proposed DWT architectures have regular structure, local communication and simple control flow, so they are very suitable for VLSI implementation and scalable filter length. In the single transform modules, the hardware utilization are 100%, so the systems consumes ultra-low power. The total data processing time of 2-D DWT with 2-samples per clock cycle can be calculated as follows:

$$\begin{aligned}&\sum_{i=0}^{j-1} 2^{-(2i+1)} \cdot (N \times N) \\ &= \frac{2^{-1}(1-2^{-2j})}{1-2^{-2}} \cdot (N \times N) \\ &= \frac{2}{3}(1-2^{-2j}) \cdot (N \times N)\end{aligned} \tag{9}$$

where $j = \log_2 N$.

The total data processing time of 2-D DWT with 4-samples per clock cycle can be calculated as follows:

$$\begin{aligned}&\sum_{i=0}^{j-1} 2^{-(2i+2)} \cdot (N \times N) \\ &= \frac{(1-2^{-2j})}{3} \cdot (N \times N)\end{aligned} \tag{10}$$

where $j = \log_2 N$.

The total data processing time of 2-D DWT with 8-samples per clock cycle can be calculated as follows:

$$\begin{aligned}&(\frac{1}{2} \cdot (\sum_{i=0}^{j-1} 2^{-(2i+2)}) - 2^{-2j}) \cdot (N \times N) + 1 \\ &= (\frac{1-4 \cdot 2^{-2j}}{6}) \cdot (N \times N) + 1\end{aligned} \tag{11}$$

where $j = \log_2 N$.

Similarly, the total data processing time of 2-D DWT processor with 16-samples per clock cycle can be calculated as follows:

$$\begin{aligned}&(\sum_{i=0}^{j-2} 2^{-(2i+4)}) \cdot (N \times N) + 1 \\ &= (2^{-4} + 2^{-6} + ... + 2^{-2j}) \cdot (N \times N) + 1 \\ &= (\frac{1-16 \cdot 2^{-2j}}{12}) \cdot (N \times N) + 1\end{aligned} \tag{12}$$

where $j = \log_2 N$.

Four high-speed and low-power architectures for 2-D DWT with a transform module have been proposed. Four proposed architectures perform compression in $2 \cdot (1-2^{-2j}) \cdot N^2/3$, $(1-2^{-2j}) \cdot N^2/3$, $(1-4 \cdot 2^{-2j}) \cdot N^2/6+1$ and $(1-16 \cdot 2^{-2j}) \cdot N^2/12+1$ processing time with 2-, 4-, 8- and 16-samples per clock cycle, respectively. They are significantly faster than conventional architectures proposed by Wu and Chen [10] [11], and Marino [12]. Table 6 depicts the comparison with previous works. In table 2, $\boldsymbol{AT^2}$ represents the system performance [11] [13] [14] [15] [16] [17], where $\boldsymbol{A}$ denotes area and $\boldsymbol{T}$ denotes time or latency (clock cycles). As can be seen, the system performance levels of the four proposed architectures are significantly better than that of previous works.

5. AREA, POWER and THROGHPUT TRADE-OFFS FOR 2-D DWT PROCESSOR

The power consumption can be represented by $\boldsymbol{AT}$ [16] [17] [18]. According to the proposed architecture with 2-samples per clock cycle, the proposed architectures with 4- 8- and 16-samples per clock cycle are reduced power consumption by 33%, but the hardware requirements are increased by 33% 167% and 400%,

respectively. The throughputs of the proposed architectures with 4-, 8- and 16-samples per clock cycle are improved by 100%, 300%, and 700%, respectively. The area, throughput and power consumption for four proposed architectures is shown in Table 2. According to Table 2, a better trade-offs between area cost and throughput is resulted from the higher sampling rate. Similarly, a better trade-offs between power consumption and area cost is also resulted from the higher sampling rate. Hence, the design requires lower power and the highest throughput, the proposed architecture with 16-samples per clock cycle, is recommended; the design requires substantially less area, the proposed architecture with 2-samples per clock cycle is suggested; and when the design requires slightly less area, lower power and higher throughput, the proposed architecture with 4- and 8-samples per clock cycle is recommended. Table 2 is a good reference for design of pipelined 2-D DWT architectures.

6. CONCLUSIONS

In this paper, throughputs and power consumptions of four proposed architectures demonstrate significant improvements. The performance levels of the proposed architectures are significantly better than those of previous works. The area, power and throughput trade-offs in the design of pipelined architectures is presented.

Filter coefficients are quantized before implementation using the biorthogonal 9/7 wavelet. The hardware arrangements are cost-effective and the systems have high speed. The proposed architectures reduce power dissipation by *m* compared with conventional architectures using multipliers in *m*-bit operand (low-power utilization).

The proposed architectures have been verified by Verilog®-HDL and implemented on FPGA. The advantages of the proposed architectures are 100% hardware utilization and ultra low-power. The proposed architectures have regular structure, simple control flow, high throughput and high scalability. Thus, they are very suitable for new-generation image compression systems, such as JPEG-2000.

REFERENCES

[1] *ITU-T Recommendation T.800. JPEG2000 image coding system – Part 1*, ITU Std., July 2002. http://www.itu.int/ITU-T/.

[2] B. E. Usevitch, “A Tutorial on Modern Lossy Wavelet Image Compression: Foundations of JPEG2000,” *IEEE Signal Processing Magazine*, Vol.18, No. 5, Sept. 2001, pp.22-35.

[3] M. Antonini, M. Barlaud, P. Mathieu, I. Daubechies, “ Image Coding Using Wavelet Transform,” *IEEE Transactions on Image Processing*, Vol.1, No.2, April 1992, pp. 205-220.

[4] K. A. Kotteri, A. E. Bell, J. E. Carletta, “Design of Multiplierless, High-Peformance, Wavelet Filter Banks with Image Compression Applications,” *IEEE Transactions on Circuits and Systems-I*, Vol. 51, No. 3, March 2004, pp.483-494.

[5] D. E. Thomas, P. H. Moorby, The Verilog Hardware Description Language, Fifth Edition, Kluwer Academic Pub. 2002.

[6] Model ModelSim Products: *http://www.model.com/products.*

[7] Synopsys FPGA Express, *http://www.synopsys.com/products.*

[8] Xilinx FPGA products, *http://www.xilinx.com/products.*

[9] S. Masud, J. V. McCanny, “Reusable Silicon IP Cores for Discrete Wavelet Transform Application,” *IEEE Transactions on Circuits and Systems-I*, Vol. 51, No. 6, June 2004, pp.1114-1124.

[10] P. –C. Wu, L. –G. Chen, “An Efficient Architecture for Two-Dimensional Discrete Wavelet Transform,” *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 11, No. 4, April 2001, pp. 536-545.

[11] P. –C. Wu, C. –T. Liu, L. –G. Chen, “An Efficient Architecture for Two-Dimensional Inverse Discrete Wavelet Transform,” *IEEE International*

Symposium on Circuits and Systems, Vol. 2, May 2002, pp. II-312-II-315.

[12] F. Marino, "Two Fast Architectures for the Direct 2-D Discrete Wavelet Transform," *IEEE Transactions on Signal Processing*, Vol. 49, No. 6, June 2001, pp. 1248-1259.

[13] S. Y. Kung, *VLSI Array Processors*, Prentice-Hall, New Jersey, USA, 1989.

[14] T. Y. Sung, C. S. Chen, "A Parallel-Pipelined Processor for Fast Fourier Transform," *The Fourth IEEE Asia-Pacific Conference on Advanced System Integrated Circuits (AP-ASIC-2004)*, Fukuoka, Japan, August 3-5, 2004, pp.194-197(10-1).

[15] T. Y. Sung, Y. S. Shieh, "A High-Speed / Ultra Low-Power Architecture for 2-D Discrete Wavelet Transform," *2005 IEEE International Conference on Systems and Signals (ICSS-2005)*, I-Shou University, Kaohsiung, Taiwan, April 28-29, 2005, pp.326-331.

[16] T. Y. Sung, Y. S. Shieh, "An Efficient Architecture for 2-D Inverse Discrete Wavelet Transform with Multiplierless Operation," *2005 IEEE International Conference on Systems and Signals (ICSS-2005)*, I-Shou University, Kaohsiung, Taiwan, April 28-29, 2005, pp.332-337.

[17] T. Y. Sung, Y. S. Shieh, " Analysis and Implementation of the Pipelined Architectures for 2-D Discrete Wavelet Transform and Its Inversion Using Direct Cascading Form," *2005 Symposium on Digital Life and Internet Technologies (Credit-2005)*, National Cheng-Kung University, Tainan, Taiwan, June 2-3, 2005.

[18] S. V. Silva, S. Bampi, "Area and Throughput Trade-Offs in the Design of Pipelined Discrete Wavelet Transform Architectures," *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition* (*DATE'05*), 2005, pp.32-37.

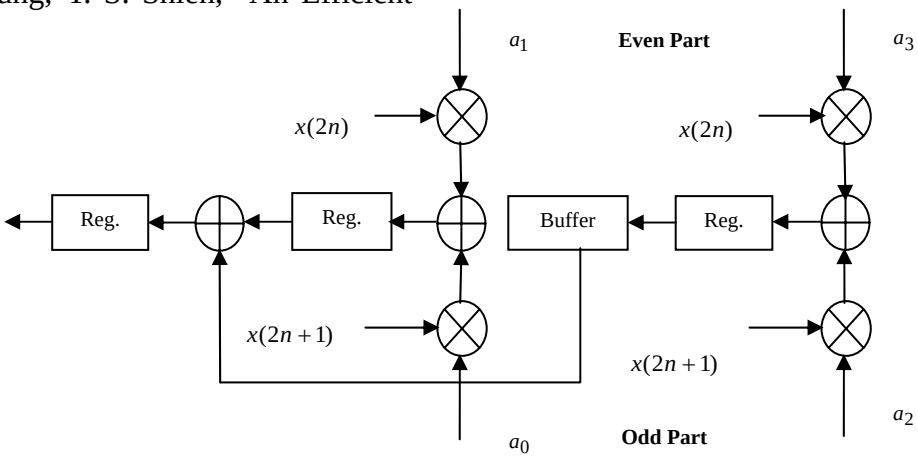

Fig. 1. The decimation filter for 1-D DWT (PE: Processing Element)

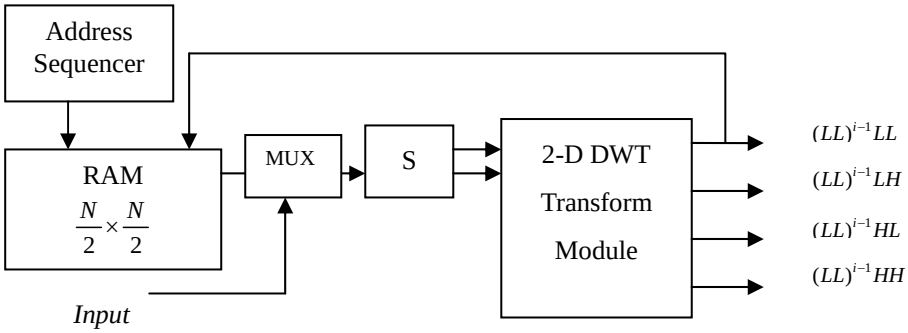

Fig. 3 The 2-D DWT processor (MUX: Multiplexer, S: Splitter)

Table 2. The hardware, throughput and power consumption for four proposed 2-D DWT architectures

Architecture	**2-samples/clock cycle [This work]**	**4- and 8-samples /clock cycle [This work]**	**16-sample/clok cycle [This work]**	**Wu & Chen [10] (1-sample per clock-cycle)**	**Marino[12] (2-samples per clock-cycle)**
Algorithm	Direct form	Direct form	Direct form	Direct form	Direct form
Latency (clock-cycle)	$2\cdot(1-2^{-2j})\cdot N^2/3$	$(1-2^{-2j})\cdot N^2/3$ $(1-4\cdot 2^{-2j})\cdot N^2/6+1$	$(1-16\cdot 2^{-2j})\cdot N^2/12+1$	$4\cdot(1-2^{-2j})\cdot N^2/3-1$	$2\cdot N^2/3$
Number of PE	6	8, 16	32	6	8
Multiplierless	Yes	Yes	Yes	No.	No.
Power Consumption ($\boldsymbol{AT}$)	AT	0.667 AT	0.667 AT	>2 AT	>2 AT
Hardware cost	Best	Better	Poor	Better	Good
Power consumption	Good	Better	Better	Poor	Poor
Throughput	High	Higher	Highest	Low	Good
System Performance ($\boldsymbol{AT}^2$)	AT^2	0.333 AT^2 0.167 AT^2	0.083 AT^2	>2.67 AT^2	>2.67 AT^2

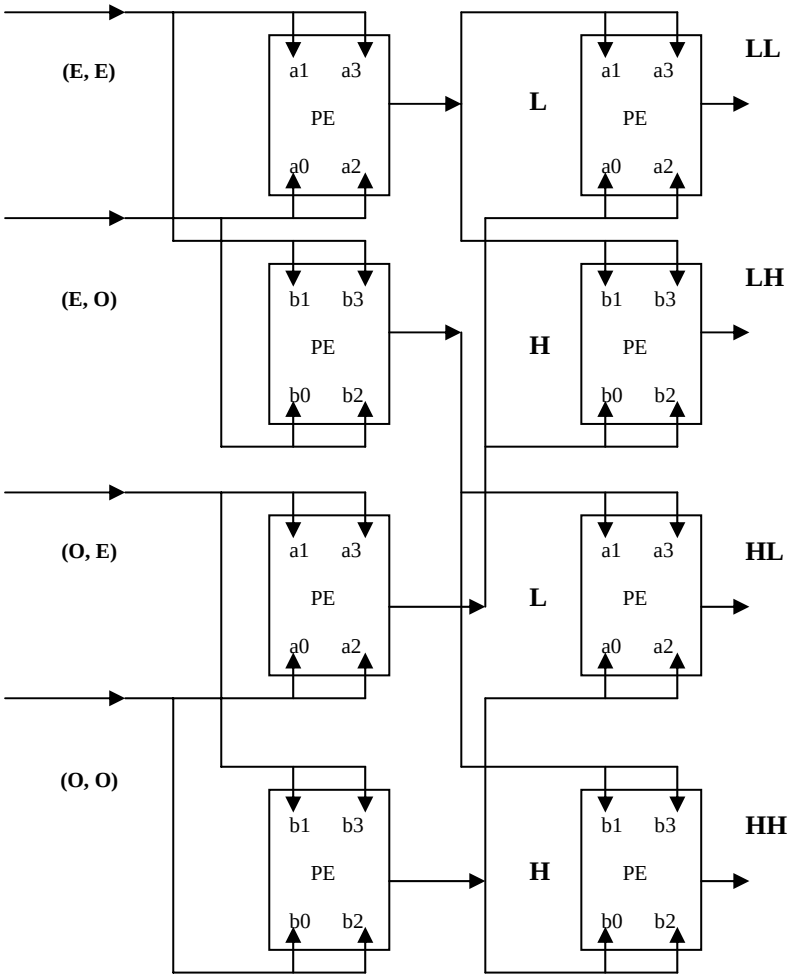

Fig. 4. The transform module with 4-sample per clock cycle for 2-D DWT processor (PE: Processing element, E: Even, O: Odd)

Table 1. Data flow of 2-D DWT processor with 16-sample per clock cycle for 3-level compression

CLK	Input (even/odd)	L	H	LL	LH	HL	HH
0	*x*(0,0),*x*(0,1),*x*(0,2),*x*(0,3) *x*(1,0),*x*(1,1),*x*(1,2),*x*(1,3) *x*(0,4),*x*(0,5),*x*(0,6),*x*(0,7) *x*(1,4),*x*(1,5),*x*(1,6),*x*(1,7)	*L*(0,0),*L*(0,1) *L*(1,0),*L*(1,1) *L*(0,2),*L*(0,3) *L*(1,2),*L*(1,3)	*H*(0,0),*H*(0,1) *H*(1,0),*H*(1,1) *H*(0,2),*H*(0,3) *H*(1,2),*H*(1,3)	*LL*(0,0) *LL*(0,1) *LL*(0,2) *LL*(0,3)	*LH*(0,0) *LH*(0,1) *LH*(0,2) *LH*(0,3)	*HL*(0,0) *HL*(0,1) *HL*(0,2) *HL*(0,3)	*HH*(0,0) *HH*(0,1) *HH*(0,2) *HH*(0,3)
1	*x*(2,0),*x*(2,1),*x*(2,2),*x*(2,3) *x*(3,0),*x*(3,1),*x*(3,2),*x*(3,3) *x*(2,4),*x*(2,5),*x*(2,6),*x*(2,7) *x*(3,4),*x*(3,5),*x*(3,6),*x*(3,7)	*L*(2,0),*L*(2,1) *L*(3,0),*L*(3,1) *L*(2,2),*L*(2,3) *L*(3,2),*L*(3,3)	*H*(2,0),*H*(2,1) *H*(3,0),*H*(3,1) *H*(2,2),*H*(2,3) *H*(3,2),*H*(3,3)	*LL*(1,0) *LL*(1,1) *LL*(1,2) *LL*(1,3)	*LH*(1,0) *LH*(1,1) *LH*(1,2) *LH*(1,3)	*HL*(1,0) *HL*(1,1) *HL*(1,2) *HL*(1,3)	*HH*(1,0) *HH*(1,1) *HH*(1,2) *HH*(1,3)
2	*x*(4,0),*x*(4,1),*x*(4,2),*x*(4,3) *x*(5,0),*x*(5,1),*x*(5,2),*x*(5,3) *x*(4,4),*x*(4,5),*x*(4,6),*x*(4,7) *x*(5,4),*x*(5,5),*x*(5,6),*x*(5,7)	*L*(4,0),*L*(4,1) *L*(5,0),*L*(5,1) *L*(4,2),*L*(4,3) *L*(5,2),*L*(5,3)	*H*(4,0),*H*(4,1) *H*(5,0),*H*(5,1) *H*(4,2),H(4,3) *H*(5,2),*H*(5,3)	*LL*(2,0) *LL*(2,1) *LL*(2,2) *LL*(2,3)	*LH*(2,0) *LH*(2,1) *LH*(2,2) *LH*(2,3)	*HL*(2,0) *HL*(2,1) *HL*(2,2) *HL*(2,3)	*HH*(2,0) *HH*(2,1) *HH*(2,2) *HH*(2,3)
3	*x*(6,0),*x*(6,1),*x*(6,2),*x*(6,3) *x*(7,0),*x*(7,1),*x*(7,2),*x*(7,3) *x*(6,4),*x*(6,5),*x*(6,6),*x*(6,7) *x*(7,4),*x*(7,5),*x*(7,6),*x*(7,7)	*L*(6,0),*L*(6,1) *L*(7,0),*L*(7,1) *L*(6,2),*L*(6,3) *L*(7,2),*L*(7,3)	*H*(6,0),*H*(6,1) *H*(7,0),*H*(7,1) *H*(6,2),*H*(6,3) *H*(7,2),*H*(7,3)	*LL*(3,0) *LL*(3,1) *LL*(3,2) *LL*(3,3)	*LH*(3,0) *LH*(3,1) *LH*(3,2) *LH*(3,3)	*HL*(3,0) *HL*(3,1) *HL*(3,2) *HL*(3,3)	*HH*(3,0) *HH*(3,1) *HH*(3,2) *HH*(3,3)
4	*LL*(0,0),*LL*(0,1) *LL*(0,2),*LL*(0,3) *LL*(1,0),*LL*(1,1) *LL*(1,2),*LL*(1,3) *LL*(2,0),*LL*(2,1) *LL*(2,2),*LL*(2,3) *LL*(3,0),*LL*(3,1) *LL*(3,2),*LL*(3,3)	*LLL*(0,0) *LLL*(1,0) *LLL*(0,1) *LLL*(1,1) *LLL*(2,0) *LLL*(3,0) *LLL*(2,1) *LLL*(3,1)	*LLH*(0,0) *LLH*(1,0) *LLH*(0,1) *LLH*(1,1) *LLH*(2,0) *LLH*(3,0) *LLH*(2,1) *LLH*(3,1)	*LLLL*(0,0) *LLLL*(0,1) *LLLL*(1,0) *LLLL*(1,1)	*LLLH*(0,0) *LLLH*(0,1) *LLLH*(1,0) *LLLH*(1,1)	*LLHL*(0,0) *LLHL*(0,1) *LLHL*(1,0) *LLHL*(1,1)	*LLHH*(0,0) *LLHH*(0,1) *LLHH*(1,0) *LLHH*(1,1)
5	LLLL(0,0),LLLL(0,1) *LLLL*(1,0),*LLLL*(1,1)	*LLLLL*(0,0) *LLLLL*(1,0)	*LLLLH*(0,0) *LLLLH*(1,0)	*LLLLLL*(0,0)	*LLLLLH*(0,0)	*LLLLHL*(0,0)	*LLLLHH*(0,0)

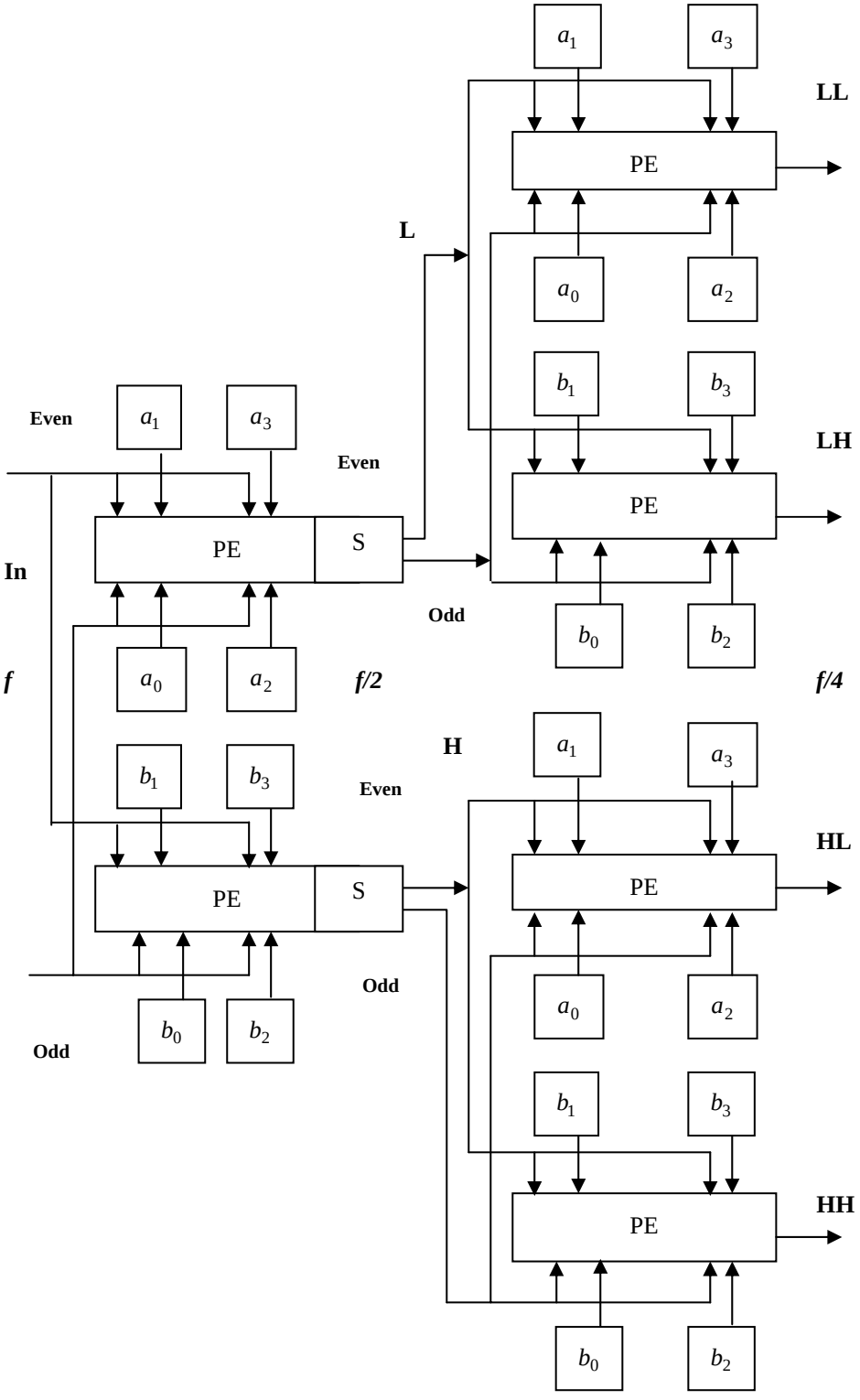

Fig. 2. The transform module with 2-sample per clock cycle for 2-D DWT processor (PE: Processing element, S: Splitter)

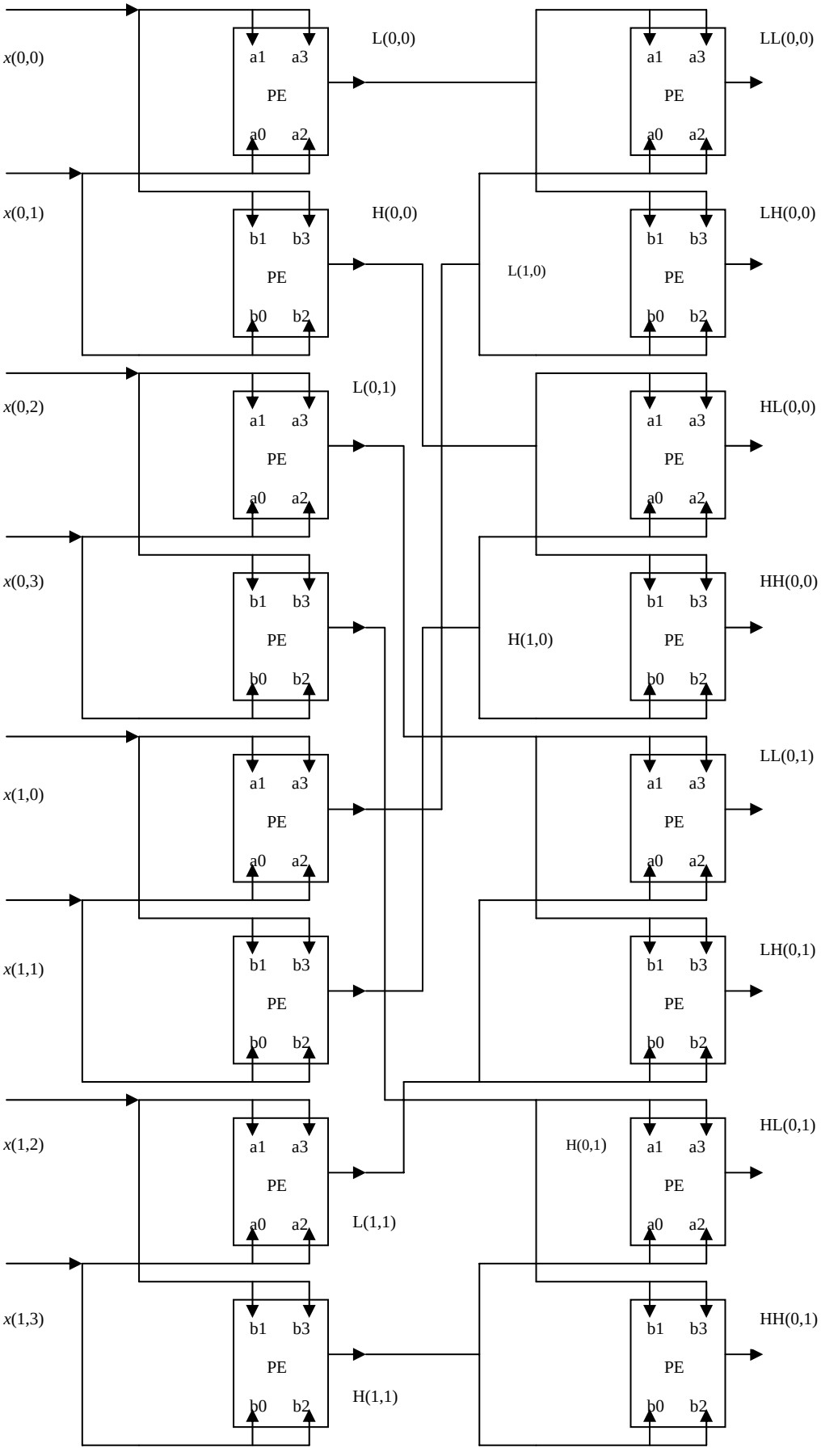

Fig. 5. The transform module with 8-sample per clock cycle for 2-D DWT processor