JOURNAL OF SEMICONDUCTOR TECHNOLOGY AND SCIENCE, VOL.12, NO.2, JUNE, 2012 http://dx.doi.org/10.5573/JSTS.2012.12.2.203

2-D Large Inverse Transform (16x16, 32x32) for HEVC (High Efficiency Video Coding)

Jong-Sik Park, Woo-Jin Nam, Seung-Mok Han, and Seongsoo Lee

***Abstract*—This paper proposes a 16x16 and 32x32 inverse transform architecture for HEVC (High Efficiency Video Coding). HEVC large transform of 16x16 and 32x32 suffers from huge computational complexity. To resolve this problem, we proposed a new large inverse transform architecture based on hardware reuse. The processing element is optimized by exploiting fully recursive and regular butterfly structure. To achieve low area, the processing element is implemented by shifters and adders without multiplier. Implementation of the proposed 2-D inverse transform architecture in 0.18 μm technology shows about 300 MHz frequency and 287 Kgates area, which can process 4K (3840x2160)@ 30 fps image.**

***Index Terms*—Large inverse transform, hardware reuse, HEVC, low area**

I. INTRODUCTION

Recent development of digital video compression technology, high quality video applications such as HDTV are popular. In the near future, next-generation video devices will have much higher definition and resolution such as UHD (Ultra High Definition) TV. In these services, multimedia data increase tremendously. For example, data amount for UHD TV is sixteen times as large as full-HD TV. It is difficult to transmit UHD resolution data to end-user over current network using current video coding standards such as H.264/AVC.

To resolve this problem, ISO-IEC/MPEG and ITU-T/VCEG recently formed the joint collaborative team on video coding (JCT-VC). It aims to develop the next-generation video coding standard called high efficiency video coding (HEVC) [1]. Main goal of HEVC is to achieve high compression, where data rate is reduced by 50% compared to H.264/AVC with same picture quality.

However, HEVC has huge computational amounts using various complex algorithms to achieve high compression efficiency. Its computation is said to be 2-4 times larger than H.264/AVC at the same picture size. Furthermore, its computation increases exponentially as HEVC supports up to 8K image.

Practically, video coding standards such as H.264/AVC should be implemented in SoC, where area reduction is a major concern [2, 3]. Especially, huge computation of HEVC results in extremely large area. To resolve this problem, new design methodology is required. In this paper, low area inverse transform architecture for HEVC is proposed. By exploiting fully recursive and regular butterfly structure, the proposed architecture achieves low area with hardware reuse, where its processing element consists of shifters, adders, and multiplexers only.

II. HEVC TRANSFORM

1. Coding Structure

In H.264/AVC, a basic coding unit is MB (Macro Block) with 16x16 pixels. But HEVC uses several basic units, i.e. CU (Coding Unit), PU (Prediction Unit), and TU (Transform Unit). CU is a basic coding unit, as MB

Manuscript received Sep. 4, 2011; revised Dec. 1, 2011.
School of Electronic Engineering, Soongsil University, Sangdo-dong, Dongjak-gu, Seoul, 156-743, Korea
E-mail : sslee@ssu.ac.kr

in H.264/AVC. CU is considered to be the fundamental square shaped unit. It has various sizes. PU is a basic prediction unit. It is defined after the last level of CU splitting. So CU can be further split into PU. TU is a basic unit for transform and quantization. Its size must be smaller than or equal to the CU size, but it can be larger than the PU size. The overall coding structure is characterized by CU, PU and TU.

By using various CU sizes, an efficient encoding for various spatial resolution and block characteristic is possible. In general, when spatial resolution is low or pixel values change significantly in local area, intra and inter prediction for small CU are more useful as shown in Fig. 1(a). When spatial resolution is high or pixel values change a little in local area, large CU can improve coding efficiency as shown in Fig. 1(b). When large CU can be used for prediction instead of a small CU, prediction error doesn't increase significantly [4].

HEVC exploits RQT (Residual Quadtree) structure [5, 6]. The prediction error is transformed and quantized based on RQT structure as shown in Fig. 2. TU size is adaptively determined based on prediction error characteristics of PUs. A PU can be further split into several TUs if the prediction errors of the split TUs in the PU are quite different. On the contrary, several PUs can be combined into a TU if the prediction errors of the combined PUs in the TU are quite similar. By

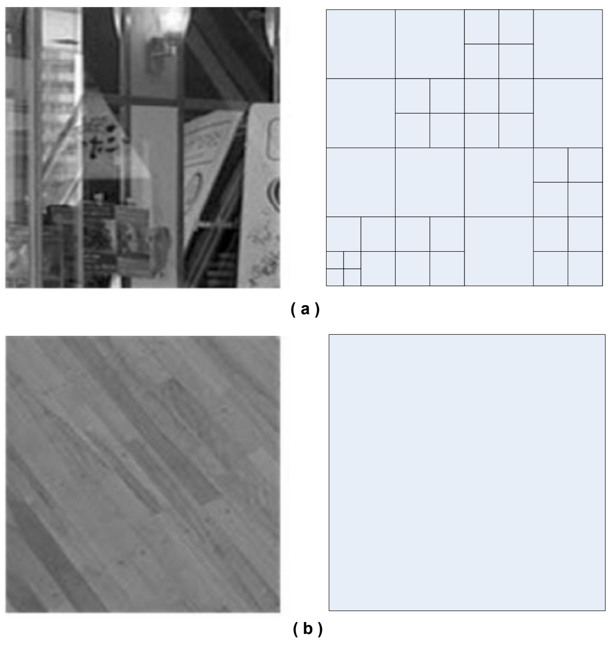


Fig. 1. CU partitioning example (a) Small CU, (b) Large CU.

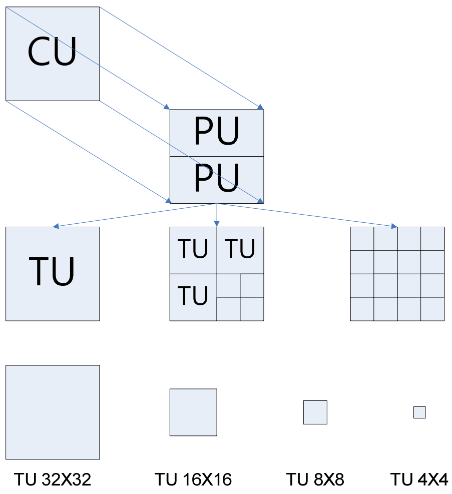


Fig. 2. RQT (residual quadtree) structure.

transforming and quantizing TUs of various sizes, the overall coding efficiency can be significantly improved.

2. Large Transform

For high resolution displays, large transform has several advantages such as better energy compaction and reduced quantization error. In HD or UHD images, most image patterns in a MB with 16x16 pixels represent a small part of objects or backgrounds, which can be described as relatively homogeneous texture patterns with little variation. Therefore, coding efficiency of high resolution video can be improved by using large transform as well as large block size [7]. To reduce complexity, HEVC large transform is based on Chen's fast DCT algorithm [8] that has fully recursive and regular butterfly structure.

III. IMPLEMENTATION

1. Hardware Reuse

Fig. 3 shows a signal flow graph of HEVC 32x32 inverse transform [5] based on Chen's fast DCT algorithm. Inputs are 32 pixels, and they are processed with 8-stage butterfly operations. So the required complexity is very high. The dotted box in Fig. 3 shows the signal flow graph of HEVC 16x16 inverse transform. As 16x16 inverse transform is also based on Chen's fast DCT algorithm, it has same butterfly structure with 6-stage but coefficient values are different. These 16x16

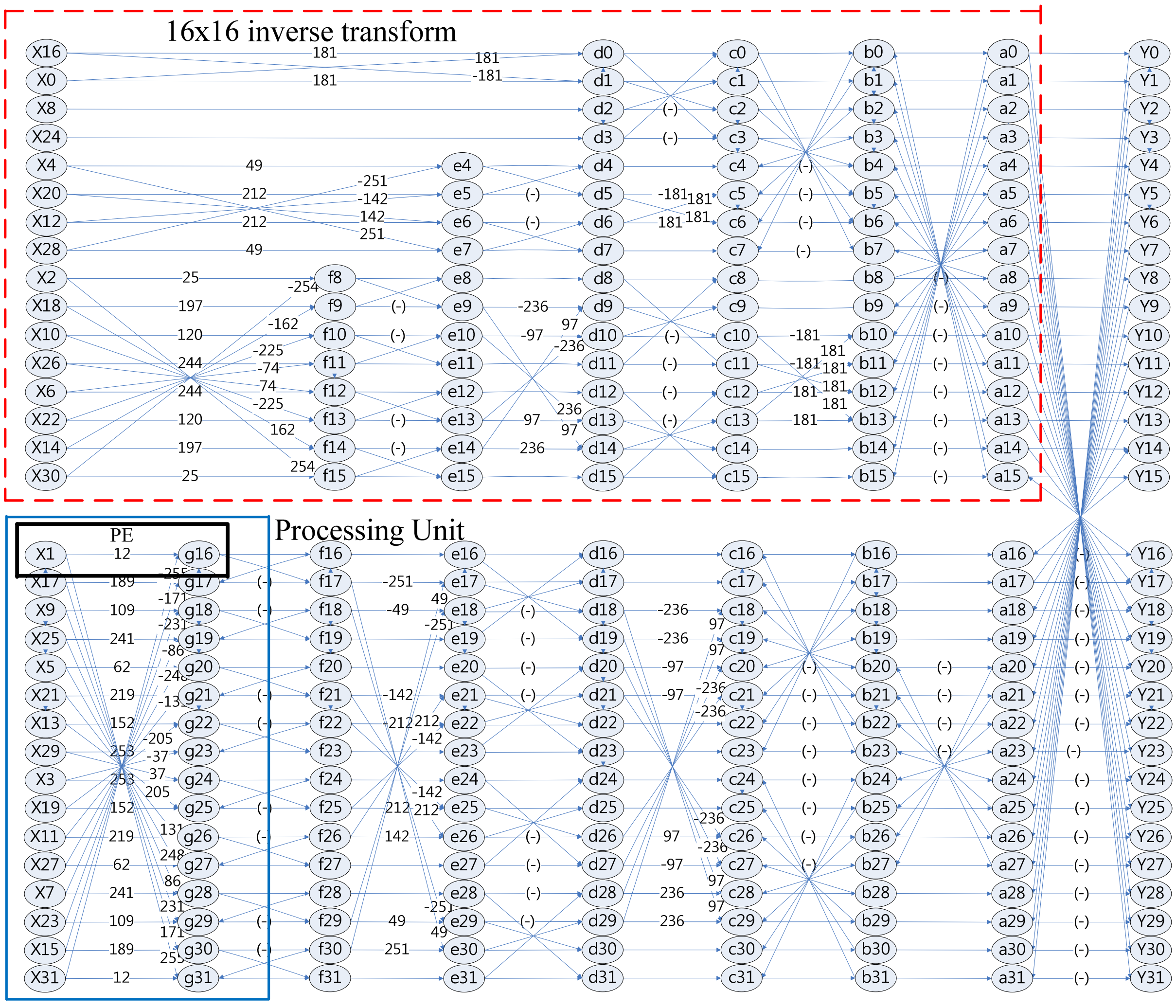


Fig. 3. Signal flow graph of HEVC 1-D large inverse transform (16x16, 32x32).

and 32x32 inverse transforms have fully recursive and regular butterfly structures. Compared with conventional 8x8 or 4x4 H.264 inverse transform, HEVC large inverse transforms (16x16 and 32x32) have extremely high complexity. This requires impractically large hardware size, so it is important to minimize hardware area. In this paper, we propose low area architecture of HEVC large inverse transform as follows. It reduces area by reusing processing elements (PE) [9]. The PE is optimized to be implemented with shifters, adders, and multiplexers only.

Fig. 4(a) shows a conventional butterfly structure. In order to compute one intermediate value, two inputs are required. One input is previous stage intermediate value on same position line. The other input is one of the previous stage intermediate value but on different position line. Calculated intermediate value of current stage is the input of next stage, recursively. Fig. 4(b) shows detailed architecture of Fig. 4(a) for 32x32 inverse transform. One input is the intermediate value on same position line. The other input is the intermediate value coming from rest 31 different position lines.

Fig. 4(b) can be implemented as Fig. 4(c) using feedback loop. This feedback loop can be exploited for hardware reuse, so the hardware size can be significantly reduced. The processing element shown in Fig. 4(c) computes bold solid line rectangle in Fig. 3. But it does not meet the required throughput if only one PE is used. Therefore, we exploited 16 PEs in a processing unit, and 16 pixels are computed by one processing unit. Note that the processing unit and the PE corresponds to the thin solid line and bold solid line rectangles in Fig. 3.

A processing unit can compute both 16x16 inverse transform and 32x32 inverse transform. If 16 PEs in a processing unit perform six stages as dotted line

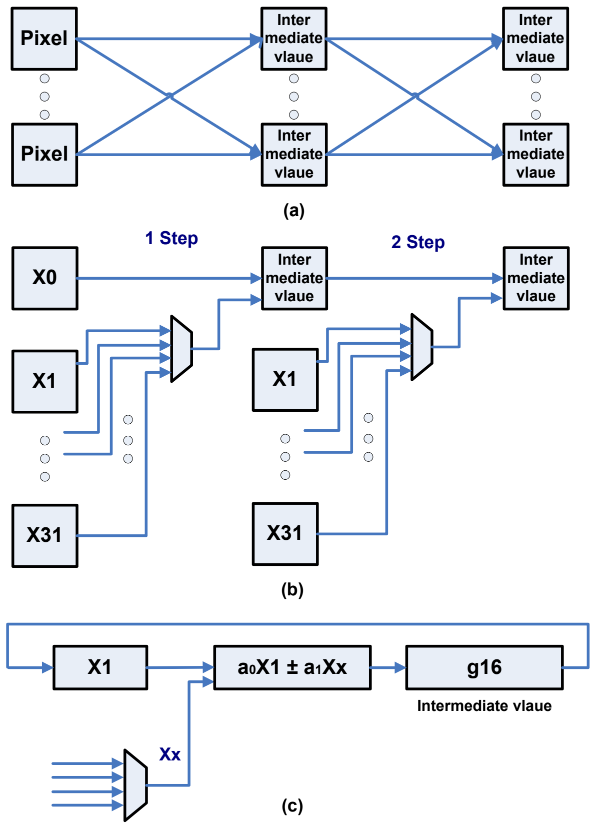


Fig. 4. (a) Conventional butterfly structure, (b) Detailed architecture of butterfly structure, (c) PE architecture using feedback loop.

rectangle in Fig. 3, 16x16 inverse transform can be computed.

32x32 inverse transform can be divided two parts. One is upper 16 outputs (Y0~Y15), and the other is lower 16 outputs (Y16~Y31). After 16 PEs in the processing unit performs six stages at upper part, calculated 16 intermediate values are stored. Then the processing unit processes seven stages at lower part and calculates other 16 intermediate values. In last stage, the processing unit calculates final 32 results from current 16 intermediate values of lower part and stored 16 intermediate values of upper part. Therefore, both 16x16 and 32x32 1-D inverse transform can be performed in one processing unit with 16 PEs. This significantly reduces hardware area.

2. PE Optimization (1): Multiplexers

One serious problem of PE architecture in Fig. 4(c) is large multiplexer size. As one input is selected from one of 31 intermediate values, a 31:1 multiplexer is required in each PE. To reduce this multiplexer size, we analyzed the position lines of PE input values from top to bottom as shown in Table 1. In 16x16 inverse transform, upper 16 outputs (Y0~Y15) are used. In 32x32 inverse transform, both upper 16 outputs (Y0~Y15) and lower 16 outputs (Y16~Y31) are used. Note that same processing units are used for upper and lower output processing, so the position lines are expressed in terms of PE number (PE0~PE15).

From Table 1, the required input values are one of only 4 intermediate values for all outputs (Y0~Y31). For

Table 1. Position lines of PE input values

Out put	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6	Stage 7	Out put	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6	Stage 7
Y0				1	3	7	15	Y16	15	1		3		7	
Y1				0	2	6	14	Y17	14	0	14	2		6	
Y2				3.	1	5	13	Y18	13	3	13	1	13	5	
Y3				2	0	4	12	Y19	12	2		0	12	4	
Y4			7	5		3	11	Y20	11	5		7	11	3	11
Y5			6	4	6	2	10	Y21	10	4	10	6	10	2	10
Y6			5	7	5	1	9	Y22	9	7	9	5		1	9
Y7			4	6		0	8	Y23	8	6		4		0	8
Y8		15	9		11		7	Y24	7	9		11		15	7
Y9		14	8	14	10		6	Y25	6	8	6	10		14	6
Y10		13	11	13	9	13	5	Y26	5	11	5	9	5	13	5
Y11		12	10		8	12	4	Y27	4	10	-	8	4	12	4
Y12		11	13		15	11	3	Y28	3	13	-	15	3	11	
Y13		10	12	10	14	10	2	Y29	2	12	2	14	2	10	
Y14		9	15	9	13		1	Y30	1	15	1	13		9	
Y15		8	14		12		0	Y31	0	14		12		8	

example, for Y0, the required position line of input values are one of 1st, 3rd, 7th, and 15th intermediate values. For Y1, the required position line of input values are one of 0st, 2nd, 6th, and 14th intermediate values. So 31:1 multiplexer can be reduced to 4:1 multiplexer for all PEs, and hardware area is significantly reduced [9].

3. PE Optimization (2): Shifters and Adders

In straightforward design, each PE requires 2 multipliers, which is impractically large. To reduce hardware area, PE can be implemented by shifters and adders without multipliers [9] based on coefficient decomposition.

Table 2 shows the coefficient decomposition. For example, intermediate value g16 is derived by g16 = (12*x1 - 255*x31)>>8. Coefficient multiplication can be decomposed by shift and addition. Coefficient 12 is decomposed into 8(<<3) +4(<<2). Coefficient 255 is decomposed into 128(<<7) + 128(<<7) - 1(<<0). In 16x16 and 32x32 inverse transform, intermediate values require two coefficients. In table 2, shift(left) and shift(right) are each coefficient decomposition for reducing multiplication. Note that (·) means the number of shift operations. Table 2 shows only 1st stage, and maximum number of shift for each PE is 4 or 5 considering all stages of 16x16 and 32x32 inverse transform. As shown Fig. 5, we exploited 4 shifters (Type A) or 5 shifters (Type B).

Table 2. Coefficient decomposition of 1st stage [9]

1 Step				
Inter Value	Operation	Coeffi-cient	Shift (left)	Shift (right)
g16	12*x1-255*x31	12, 255	(2) 3, 2	(3) 7, 7, -0
g17	189*x17-171*x15	189, 171	(4) 7, 6, -1, -0	(5) 7, 6, -4, -2, -0
g18	109*x9-231*x23	109, 231	(4) 7, -4, -1, -0	(4) 8, -4, -3, -0
g19	241*x25-86*x7	241, 86	(4) 7, 7, -4, 0	(4) 7, -5, -3, -1
g20	62*x5-248*x27	62, 248	(2) 6, -1	(3) 7, 7, -3
g21	219*x21-131*x11	219, 131	(4) 8, -5, -2, -0	(3) 7, 1, 0
g22	152*x13-205*x19	152, 205	(3) 7, 4, 3	(5) 7, 6, 4, -2, 0
g23	253*x29-37*x3	253, 37	(4) 7, 7, -1, -0	(3) 5, 2, 0
g24	253*x3+37*x29	253, 37	(4) 7, 7, -1, -0	(3) 5, 2, 0
g25	152*x19+205*x13	152, 205	(3) 7, 4, 3	(5) 7, 6, 4, -2, 0
g26	219*x11+131*x21	219, 131	(4) 8, -5, -2, -0	(3) 7, 1, 0
g27	62*x27+248*x5	62, 248	(2) 6, -1	(3) 7, 7, -3
g28	241*x7+86*x25	241, 86	(4) 7, 7, -4, 0	(4) 7, -5, -3, -1
g29	109*x23+231*x9	109, 231	(4) 7, -4, -1, -0	(4) 8, -4, -3, -0
g30	189*x15+171*x17	189, 171	(4) 7, 6, -1, -0	(5) 7, 6, -4, -2, -0
g31	12*x31+255*x1	12, 255	(2) 3, 2	(3) 7, 7, -0

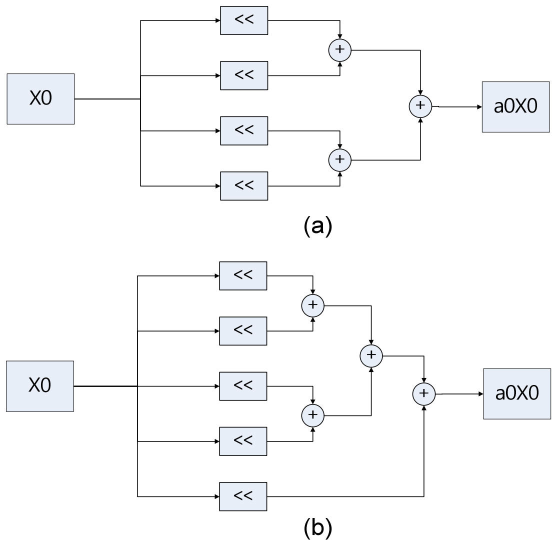


Fig. 5. Type of Shifters and adders (a) Type A, (b) Type B.

4. 2-D Inverse Transform

For high resolution such as 4k and 8k images, 32x32 2-D inverse transform is required. In this case, 1 processing unit with 16 PEs is not enough for throughput requirement. Therefore, we used two 1-D processing units in pipelined manner. As shown Fig. 6, when one processing unit executes 2-D transform for current TU#0, the other processing unit can execute 1-D transform for next TU#1. So total operation cycle can be reduced. However, in this case, two 32x32 buffers are required – one for TU#0 outputs and the other for TU#1 outputs. Furthermore, these buffers should perform transpose operation.

To resolve this problem, we used 1 32x32 transpose buffer, shared with 2 processing units in common. As shown in Fig. 6 ⓐ, when 1-D transform of current TU#0 is finished, 1-D all outputs are stored in transpose buffer. As shown in Fig. 6 ⓑ, upper stored values in transpose buffer are inputs of first 2-D transform. As shown in Fig. 6 ⓒ, if first 2-D transform starts, upper stored values in transpose buffer is not necessary. Therefore, first output of next TU#1 can be stored in upper transpose buffer as shown in Fig. 6 ⓓ. When all 2-D transform of current

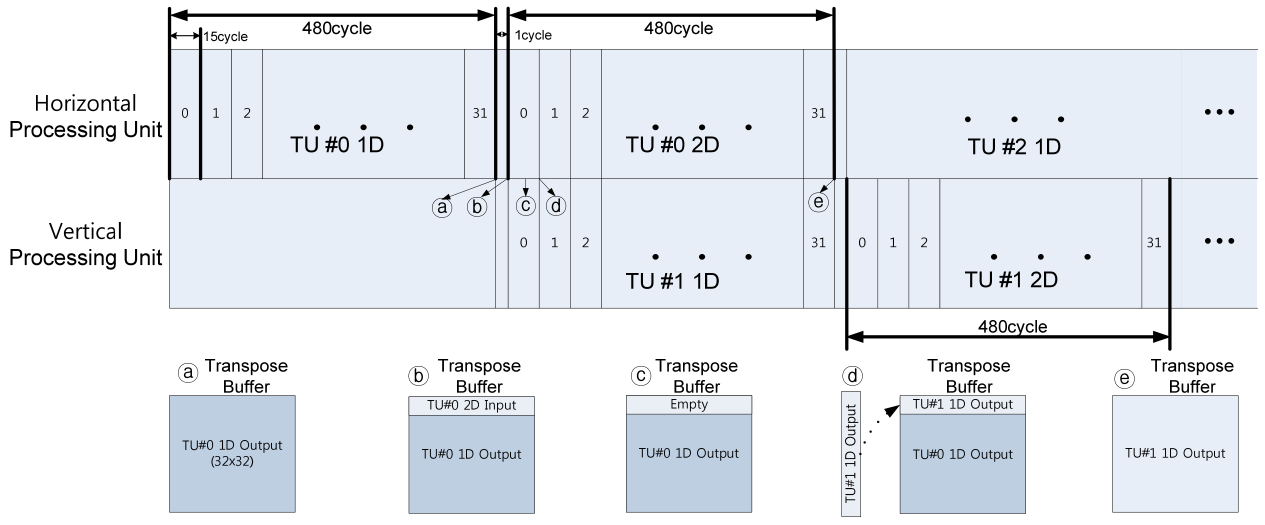


Fig. 6. 2-D inverse transform.

TU#0 is finished, 1-D all outputs of next TU#1 are stored in transpose buffer as shown in Fig. 6 ⓔ.

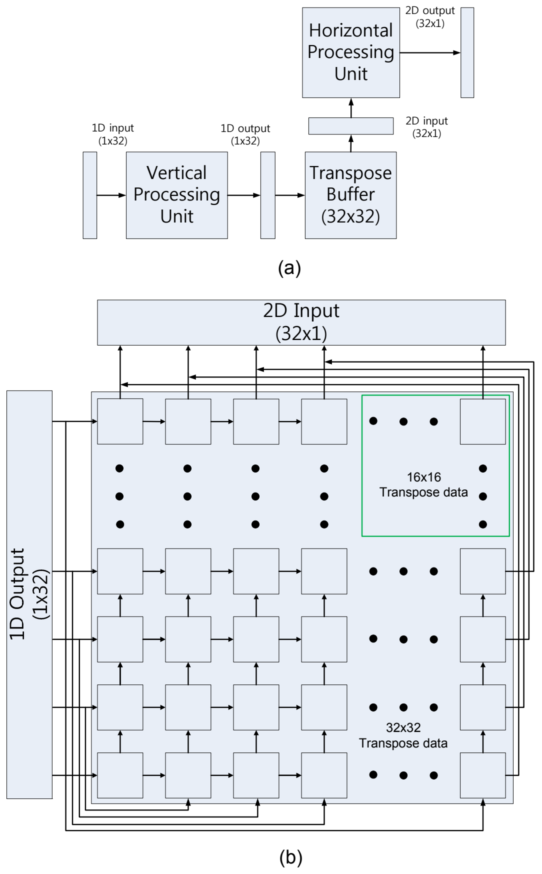
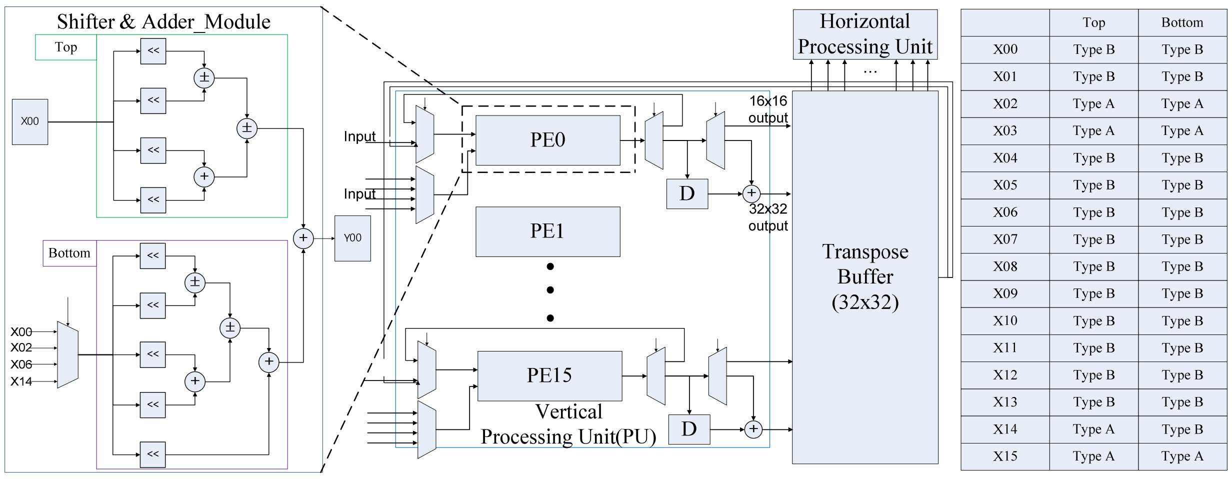


Fig. 7. (a) Relation of between processing unit and transpose buffer, (b) Transpose buffer architecture.

Fig. 7(a) shows operation of processing units and transpose buffer. Vertical processing unit operates 1-D transform, and horizontal processing unit operates 2-D transform. Whenever 1-D transpose data (1x32) of current TU are calculated, transpose data shifts right as shown in Fig. 7(b). Whenever 2-D transpose data (32x1) of current TU are calculated, transpose data shifts up. Proposed architecture supports both 16x16 and 32x32 inverse transform. So transpose data (16x16) are stored right-top of transpose buffer when the proposed architecture performs 16x16 inverse transform.

5. Proposed Architecture

Fig. 8 shows the proposed 2-D HEVC 16x16 and 32x32 inverse transform architecture. It consists of 2 processing units (vertical and horizontal processing units) and 1 transpose buffer. As mentioned above, the processing unit operates each stage recursively. Inputs of processing units are X00, X01, …, X15 which are input values or derived intermediate values. Outputs of processing units are Y00, Y01, …, Y15 which are derived intermediate values or output values. In other words, outputs of current stage are inputs of next stage. Register D is used for butterfly operation of last stage of 32x32 inverse transform. Types of top and bottom shifter/adder modules in each PE are described in the right table of Fig. 8.



	Top	Bottom
X00	Type B	Type B
X01	Type B	Type B
X02	Type A	Type A
X03	Type A	Type A
X04	Type B	Type B
X05	Type B	Type B
X06	Type B	Type B
X07	Type B	Type B
X08	Type B	Type B
X09	Type B	Type B
X10	Type B	Type B
X11	Type B	Type B
X12	Type B	Type B
X13	Type B	Type B
X14	Type A	Type B
X15	Type A	Type A

Fig. 8. Architecture of the proposed HEVC large inverse transform IP.

IV. RESULTS

The proposed 2-D HEVC large inverse transform (16x16, 32x32) is implemented by 0.18 μm CMOS technology. Total gate count of proposed architecture is about 287K gates (2-NAND equivalent). Among them, transpose buffer is about 183K gates.

Since 32x32 inverse transform architecture is not proposed in previous researches yet, we compared the proposed architecture and other 1-D 8x8 inverse transform architectures, as shown in Table 3. Considering 32x32 block size (16 times larger than 8x8), proposed architecture shows quite small gate counts. Fig. 9 shows the layout of the proposed HEVC large inverse transform IP. Its maximum operating frequency is 300 MHz.

Execution cycle of proposed architecture for 32x32 block is 481 cycles, i.e. 2.1289 pixels/cycle, as shown Fig. 6. Required throughput for 4k(3840x2160)@30 fps image is 248,832,000 pixels/s. Therefore, required operating frequency is 248,832,000/2.1289 = 117 MHz. However, in many video decoders, there may be some data stall to wait data from and to its connecting IPs such as motion compensation, intra prediction, and inverse quantization IPs. Even the proposed architecture is stalled in 50% of its operation time, it can easily support 4K@30 fps image, since the required operating frequency is 234MHz.

In H.264 decoders, inverse transform IP is small and fast. Therefore, in many cases, it is directly connected to other IPs without interface buffer. Note that interface buffer greatly reduces data stall, but it occupies much additional area. However, 32x32 HEVC inverse

Table 3. Comparison of 1-D transform architectures

Researches	Tseng [10]	Sung [11]	Chen [12]	Lai [13]	Proposed
Algorithm	SA-DCT	COR DIC	Scalable-DA	NEDA	HEVC Large Transform
Transform size	8x8	8x8	8x8	8x8	16x16 32x32
Technology	0.35 μm	0.18 μm	0.13 μm	90 nm	0.18 μm
Multipliers or ROMs	Yes	Yes	Yes	No	No
Gate counts for 1-D transform	160K	N/A	15.2K	11.6K	52.3K

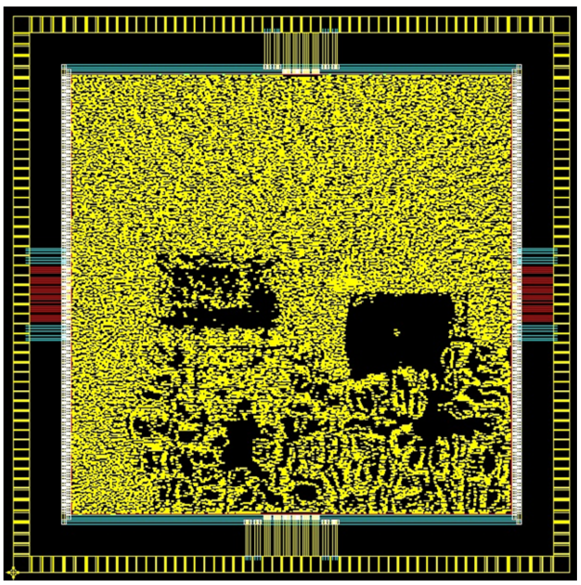


Fig. 9. Layout of the proposed HEVC large inverse transform IP.

transform IP is quite large, so it is advantageous to use interface buffer to speed up. When the proposed architecture is stalled in 20% of its operation time, the required operating frequency for 4k@60 fps image is 292 MHz. Therefore, the proposed architecture can support 4K@60 fps image by exploiting interface buffer.

V. CONCLUSIONS

In this paper, we proposed 2-D large inverse transform architecture for HEVC. To reduce hardware area, we exploited hardware reuse and optimized adder/shifter/ multiplexer-only PE architecture. Implementation of the proposed 2-D inverse transform architecture in 0.18 μm technology shows about 300 MHz frequency and 287 Kgates area, which can process 4K@30 fps image.

ACKNOWLEDGEMENTS

This work was supported by Industrial Strategic Technology Development Program funded by the Ministry of Knowledge Economy, Korea (10039239).

REFERENCES

[1] K. Ugur, K. R. Andersson, A. Fuldseth, "Low Complexity Video Coding and the Emerging HEVC Standard", *Proceedings of Picture Coding Symposium (PCS)*, pp.474-477, Dec., 2010.

[2] N. Hirai, T. Song, Y. Liu, and T. Shimamoto, "A Novel Spiral-Type Motion Estimation Architecture for H.264/AVC," *Journal of Semiconductor Technology and Science*, Vol.10, No.1, pp.37-44, Mar., 2010.

[3] C. Lee, "Smart Bus Arbiter for QoS Control in H.264 Decoders", *Journal of Semiconductor Technology and Science*, Vol.11, No.1, pp.33-39, Mar., 2011.

[4] Chanwon Seo, Jongki Han, "Video Coding Performance for Hierarchical Coding Block and Transform Block Structures", *Korea Society Broading Engineers Magazine*, Vol.15, No.4, pp.23-34, Dec., 2010.

[5] Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, "WD2: Working Draft 2 of High-Efficiency Video Coding", Jan., 2011

[6] Martin Winken, Detlev Marpe, Heiko Schwarz, "Highly Efficient Video Coding based on Quadtree Structures, Improved Motion Compensation, and Probability Interval Partitioning Entropy Coding", *Proceedings of ITG Conference on Electronic Media Technology (CEMT)*, pp.1-6, Mar., 2011

[7] K. McCann, W. J. Han, I. K. Kim, J. H. Min, "Samsung's response to the call for proposals on video compression technology", 1st JCT-VC meeting , No.JCTVC-A124, Apr., 2010.

[8] W.-H. Chen, C. H. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform", *IEEE Transactions on Communications,* vol.25, No.9, pp.1004-1009, Sep., 1977.

[9] J. Park, W. Nam, S. Han, and S. Lee, "High Efficiency Video Coding (HEVC) 16x16 & 32x32 Inverse Transform IP Design for Large-Scale Displays," *Proceedings of Internation Technical Conference on Circuits/Systems, Computers, and Communications (ITC-CSCC)*, pp.153-155, Jun., 2011.

[10] P. C. Tseng, C. T. Haung, and L. G. Chen, "Reconfigurable discrete cosine transform processor for object-based video signal processing," *Proceedings of International Symposium on Circuits and Systems (ISCAS)*, pp.353-356, May, 2004.

[11] T. Y. Sung, Y. S. Shieh, C. W Yu, and H, C. Hsin, "High-Efficiency and Low Power Architectures for 2-D DCT and IDCT Based on CORDIC Rotation," *Proceedings of International Conference on Parallel and Distributed Computing, Applications, and Technologies (PDCAT)*, pp.191-196, Dec., 2006.

[12] J. W. Chen, K. Hung, J. S. Wang, and J. I. Guo, "A Performance Aware IP Core Design for Multi-mode Transform Coding Using Scalable-DA Algorithm," *Proceedings of International Symposium on Circuits and Systems (ISCAS)*, pp.21-24, May, 2006.

[13] Y.K. Lai, Y.F. Lai, "A reconfigurable IDCT architecture for universal video decoders", *IEEE Transactions on Consumer Electronics*, Vol.56, No.3, pp.1872-1879, Aug., 2010.

Jong-Sik Park received B.S. and M.S. degrees in E.E. from Soongsil University, Korea, in 2002 and 2007, respectively. He is currently pursuing Ph.D. degree in E.E. from Soongsil University, Korea. His research interests include SoC implementation of digital system, multimedia system (H.264, HEVC), low power system, and multi-view video coding.

Woo-Jin Nam received B.S. degree in E.E. from Soongsil University, Korea, in 2011. He is currently pursuing M.S degree in E.E. Soongsil University, Korea. His research interests include high efficiency video coding, battery lifetime enhancement technology, and low-power system architecture.

Seung-Mok Han received B.S. degree in E.E. from Soongsil University, Korea, in 2011. He is currently pursuing M.S degree in E.E. Soongsil University, Korea. His research interests include high efficiency video coding, semiconductor, digital circuit, multimedia, and ultra-low-power analog circuit.

Seongsoo Lee received B.S, M.S, and Ph.D degrees in E.E. from Seoul National University, Korea in 1991, 1993, and 1998, respectively. In 1998-2000, he was a research associate in Institute of Industrial Science, University of Tokyo, Japan. In 2000-2002, he was a research professor in Department of Electronic Engineering, Ewha Womans University, Korea. He joined School of Electronic Engineering at Soongsil University, Korea in 2002, where he is currently an associate professor. His research interests include low-power SoC, multimedia SoC, and battery management SoC.