High-Performance Architecture of Transform Circuit for Multi-standard Video CODEC

Seonyoung Lee[1] and Kyeongsoon Cho[2]
Department of Electronics and Information Engineering, Hankuk University of Foreign Studies
89 Wangsan-ri, Mohyun-myun, Cheoin-gu, Yongin-si, Gyeonggi-do, Republic of Korea
E-mail: [1]drleesy@hufs.ac.kr, [2]kscho@hufs.ac.kr

Abstract: This paper presents the architecture of transform circuit that can support multiple video CODEC standards such as JPEG, MPEG-1/2/4, H.264 and VC-1. The proposed architecture exploits the similarity of 4-point and 8-point DCT's based on the permutation matrices. Since our circuit accepts the transform coefficients from the users, it can be extended very easily to cover any kind of DCT-based transforms for future standards. We described the proposed transform circuit at RTL and verified its operation on FPGA board.

1. Introduction

Portable multimedia systems become more popular as better multimedia services are available. It will be too expensive to have dedicated video encoder/decoder (CODEC) chips for each standard in the multimedia system. The solution to this problem is to have a single chip that can support multiple standards. This paper proposes a new high-performance circuit architecture that can efficiently perform all kinds of DCT-based transforms for the standards such as JPEG[1], MPEG-1/2, MPEG-4[2], H.264[3] and VC-1[4]. The proposed architecture consists of four transform array elements and multi-purpose buffers. Each array element performs one-dimensional (1D) transform at a rate of four pixels per cycle. Two-dimensional (2D) transform is achieved by using multi-purpose buffers as transpose buffers. These buffers are also used to reorder the data to perform permutation operations. To support various transforms, our circuit allows the user to load transform coefficients from external sources. The proposed architecture can be extended very easily to cover any kind of DCT-based transforms for future standards. The number of logic gates used in our circuit is 30,173 and the maximum operating frequency is 116.8MHz when synthesized using 130nm standard cells.

2. Proposed Architecture of High-Performance Transform Circuit

2. 1 4-point and 8-point DCT's

$$X_{uv} = \frac{1}{4} c(u)c(v) \sum_{m=0}^{7}\sum_{n=0}^{7} x_{mn} \cos\left[\frac{(2m+1)u\pi}{16}\right] \cos\left[\frac{(2n+1)v\pi}{16}\right] \quad (1)$$

$$c(n) = \begin{cases} 1/\sqrt{2} & , when\ n = 0 \\ 1 & , otherwise \end{cases}, \ u, v = 0, 1, \ldots, 7.$$

Most of video CODEC's are using two kinds of discrete cosine transform (DCT): 8-point and 4-point DCT's. Equation (1) defines the 2D 8-point DCT, which is used in JPEG, and MPEG-1/2/4.

$$\begin{aligned} E_{MxN} &= (W_{MxM} \cdot X_{MxN} + 4) >> 3 \\ R_{MxN} &= (E_{MxN} \cdot W_{NxN}^{T} + C_M \cdot 1_N + 64) >> 7 \end{aligned} \quad (2)$$

VC-1 includes both of 8-point and 4-point DCT's, defined by (2). E_{MxN} is the intermediate matrix obtained by 1D DCT and R_{MxN} is the resultant matrix of the 2D DCT. X_{MxN} is the input matrix of the DCT. The matrices W_{MxM} and W_{NxN} contain the DCT coefficients of VC-1. C_M is a column vector whose dimension is M and 1_N is a row vector whose dimension is N. The values of M and N are either 4 or 8. H.264 requires only 4-point DCT for residual data, luma DC data, and chroma DC data. All the coefficients in the H.264 DCT matrices are either ±1 or ±1/2.

As discussed above, the DCT for all video CODEC's is either 4-point or 8-point DCT. The coefficients of 8-point DCT for JPEG, MPEG-1/2/4 and VC-1 can be classified into seven different coefficients (*a*, *b*, *c*, *d*, *e*, *f* and *g*) as shown in (3).[5] The coefficients of 4-point DCT for H.264 and VC-1 can be classified into three different coefficients (*h*, *i* and *j*) as shown in (4).

$$C_{8p} = \begin{pmatrix} a & a & a & a & a & a & a & a \\ b & c & d & e & -e & -d & -c & -b \\ f & -g & g & -f & -f & g & -g & f \\ c & -e & -b & -d & d & b & e & -c \\ a & -a & -a & a & a & -a & -a & a \\ d & -b & e & c & -c & -e & b & -d \\ -g & -f & f & g & g & f & -f & -g \\ e & -d & c & -b & b & -c & d & -e \end{pmatrix} \quad (3)$$

$$C_{4p} = \begin{pmatrix} h & h & h & h \\ i & j & -j & -i \\ h & -h & -h & h \\ j & -i & i & -j \end{pmatrix} \quad (4)$$

2. 2 4-point DCT with permutation matrices

The coefficient matrix of 4-point DCT can be reordered by the two permutation matrices defined in (5).[6] These permutation matrices are unitary matrices which satisfy $P_{4c} P_{4c}^{T} = P_{4c}^{T} P_{4c} = I_4$, and $P_{4r} P_{4r}^{T} = P_{4r}^{T} P_{4r} = I_4$.

$$P_{4c} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad P_{4r} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (5)$$

We define $\tilde{C}_{4p}$ as (6) using the permutation matrices and it results in (7).

$$P_{4c}\, C_{4p}\, P_{4r} = \tilde{C}_{4p} \quad (6)$$

$$\tilde{C}_{4p} = \begin{pmatrix} h & h & h & h \\ h & -h & -h & h \\ i & j & -j & -i \\ j & -i & i & -j \end{pmatrix} \quad (7)$$

The signal flow graph (SFG) in Fig. 1 represents the operations of $\tilde{C}_{4p} U = V$. Equation (8) shows the relationship between C_{4p} and $\tilde{C}_{4p}$.

$$C_{4p} = P_{4c}\, \tilde{C}_{4p}\, P_{4r} \quad (8)$$

The operations of 4-point forward DCT are given by (9).

$$Y_{4p} = C_{4p}\, X_{4p}\, C_{4p}^{\mathrm{T}} = (P_{4c}\, \tilde{C}_{4p}\, P_{4r})\, X_{4p}\, (P_{4c}\, \tilde{C}_{4p}\, P_{4r})^{\mathrm{T}} \quad (9)$$

Equation (10) is derived from $(P_{4c}\, \tilde{C}_{4p}\, P_{4r})^{\mathrm{T}} = P_{4r}^{\mathrm{T}}\, \tilde{C}_{4p}^{\mathrm{T}}\, P_{4c}^{\mathrm{T}}$, $P_{4r}^{\mathrm{T}} = P_{4r}$ and $P_{4c}^{\mathrm{T}} = P_{4c}$.

$$Y_{4p} = P_{4c}\, \tilde{C}_{4p}\, P_{4r}\, X_{4p}\, P_{4r}\, \tilde{C}_{4p}^{\mathrm{T}}\, P_{4c} \quad (10)$$

If we define Z_{4p} as $P_{4r}\, X_{4p}\, P_{4r} = Z_{4p}$, (10) can be rewritten as (11).

$$Y_{4p} = P_{4c}\, (\tilde{C}_{4p}\, Z_{4p}\, \tilde{C}_{4p}^{\mathrm{T}})\, P_{4c} = P_{4c}\, Y_{4p}^{'}\, P_{4c} \quad (11)$$

where $\tilde{C}_{4p} Z_{4p} \tilde{C}_{4p}^{\mathrm{T}} = Y_{4p}^{'}$. Note that Z_{4p} and Y_{4p} can be obtained by reordering the elements in X_{4p} and $Y_{4p}^{'}$ according to the permutation matrices without any computational effort.

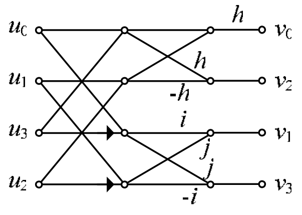


Figure 1. Signal flow graph for $\tilde{C}_{4p}$.

2. 3 8-point DCT with permutation matrices

We applied the concept of using permutation matrices to represent the 8-point DCT. We defined the permutation matrices for 8-point DCT as (12) and (13). They are unitary matrices which satisfy $P_{8c}\, P_{8c}^{\mathrm{T}} = P_{8c}^{\mathrm{T}}\, P_{8c} = I_8$, and $P_{8r}\, P_{8r}^{\mathrm{T}} = P_{8r}^{\mathrm{T}}\, P_{8r} = I_8$.

$$P_{8c} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad (12)$$

$$P_{8r} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (13)$$

We define $\tilde{C}_{8p}$ as (14) using the permutation matrices P_{8c} and P_{8r}.

$$\tilde{C}_{8p} = \begin{pmatrix} a & a & a & a & a & a & a & a \\ f & g & g & f & -f & -g & -g & -f \\ b & c & -c & -b & -e & -d & d & e \\ c & -e & e & -c & d & b & -b & -d \\ a & -a & -a & a & a & -a & -a & a \\ g & -f & -f & g & -g & f & f & -g \\ d & -b & b & -d & -c & -e & e & c \\ e & -d & d & -e & b & -c & c & -b \end{pmatrix} \quad (14)$$

$\tilde{C}_{8p}$ can be expressed as shown in (15) using four matrices $\tilde{C}_{00}$, $\tilde{C}_{01}$, $\tilde{C}_{10}$ and $\tilde{C}_{11}$ in (16).

$$\tilde{C}_{8p} = \begin{bmatrix} \tilde{C}_{00} & \tilde{C}_{01} \\ \tilde{C}_{10} & \tilde{C}_{11} \end{bmatrix} \quad (15)$$

The operations for the matrices in (16) can be represented by the SFG's as illustrated in Fig. 2. All the graphs are the same except the coefficient values. A common operation unit can be constructed for $\tilde{C}_{00}$, $\tilde{C}_{01}$, $\tilde{C}_{10}$ and $\tilde{C}_{11}$ based on the observation that they have the same structure. We can find that $\tilde{C}_{4p}$ in Fig. 1 and $\tilde{C}_{00}$ in Fig. 2 (a) are very similar. We merged the SFG's in Fig. 1 and Fig. 2 into the graphs in Fig. 3 to share more transform operations. Hence there is a possibility of sharing between 4-point and 8-point DCT's.

$$\tilde{C}_{00}=\begin{pmatrix} a & a & a & a \\ f & g & g & f \\ b & c & -c & -b \\ c & -e & e & -c \end{pmatrix} \quad \tilde{C}_{01}=\begin{pmatrix} a & a & a & a \\ -f & -g & -g & -f \\ -e & -d & d & e \\ d & b & -b & -d \end{pmatrix}$$
$$\tilde{C}_{10}=\begin{pmatrix} a & -a & -a & a \\ g & -f & -f & g \\ d & -b & b & -d \\ e & -d & d & -e \end{pmatrix} \quad \tilde{C}_{11}=\begin{pmatrix} a & -a & -a & a \\ -g & f & f & -g \\ -c & -e & e & c \\ b & -c & c & -b \end{pmatrix} \tag{16}$$

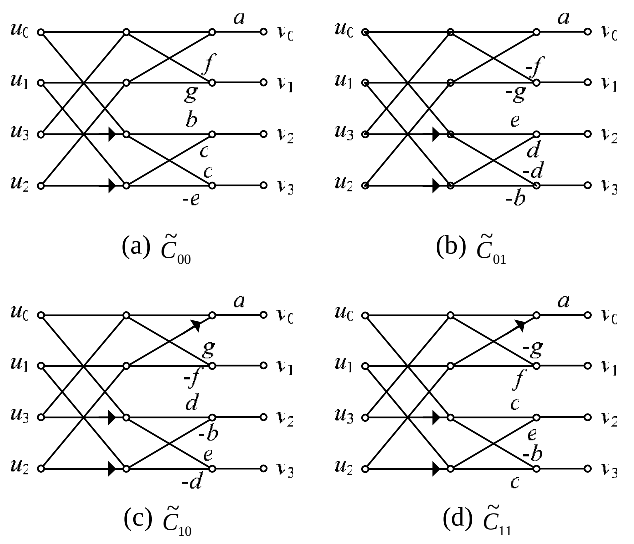
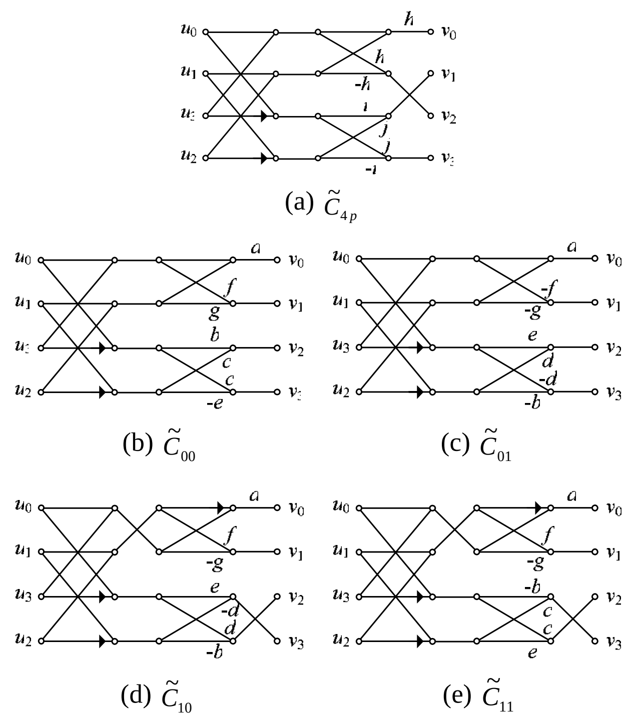

(a) $\tilde{C}_{00}$ (b) $\tilde{C}_{01}$ (c) $\tilde{C}_{10}$ (d) $\tilde{C}_{11}$

Figure 2. Signal flow graphs for $\tilde{C}_{00}$, $\tilde{C}_{01}$, $\tilde{C}_{10}$ and $\tilde{C}_{11}$.

The 1D 8-point forward DCT can be expressed by (17) using several 4x4 matrices. $\tilde{C}_{8p}$, Z_{8p} are 8x8 matrices, and $\tilde{C}_{00}$, $\tilde{C}_{01}$, $\tilde{C}_{10}$, $\tilde{C}_{11}$, Z_{00}, Z_{01}, Z_{10}, Z_{11} are 4x4 matrices.

$$\begin{aligned}\tilde{C}_{8p}\,Z_{8p} &= \begin{bmatrix}\tilde{C}_{00} & \tilde{C}_{01} \\ \tilde{C}_{10} & \tilde{C}_{11}\end{bmatrix}\begin{bmatrix}Z_{00} & Z_{01} \\ Z_{10} & Z_{11}\end{bmatrix} \\ &= \begin{bmatrix}\tilde{C}_{00}Z_{00}+\tilde{C}_{01}Z_{10} & \tilde{C}_{00}Z_{01}+\tilde{C}_{01}Z_{11} \\ \tilde{C}_{10}Z_{00}+\tilde{C}_{11}Z_{10} & \tilde{C}_{10}Z_{01}+\tilde{C}_{11}Z_{11}\end{bmatrix}\end{aligned} \tag{17}$$

The multiplication of the 8x8 matrices is performed by multiplying 4x4 matrices and adding the results. Hence the intermediate results $\tilde{C}_{00}Z_{00}$, $\tilde{C}_{10}Z_{00}$, $\tilde{C}_{00}Z_{01}$, $\tilde{C}_{10}Z_{01}$ will be stored in the buffers and then added to $\tilde{C}_{01}Z_{10}$, $\tilde{C}_{11}Z_{10}$, $\tilde{C}_{01}Z_{11}$, $\tilde{C}_{11}Z_{11}$ when we design the circuit.


(a) $\tilde{C}_{4p}$

(b) $\tilde{C}_{00}$ (c) $\tilde{C}_{01}$

(d) $\tilde{C}_{10}$ (e) $\tilde{C}_{11}$

Figure 3. Merged signal flow graphs for sharing operations.

3. Circuit Implementation and Results

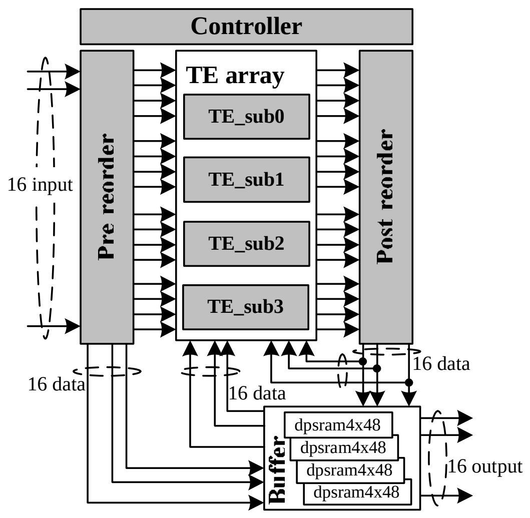


Figure 4. Block diagram of proposed transform circuit.

Fig. 4 illustrates the architecture of the proposed transform circuit based on the equations described in the previous section. The proposed circuit is composed of Pre-reorder, TE array, Post-reorder, Buffer and Controller. Our 1D transform circuit accepts 16 pixels per cycle as inputs and provides 16 pixels per cycle as outputs. Pre-reorder block accepts 16 pixels per cycle and performs the permutation operations $P_{4r}\ X_{4p}\ P_{4r} = Z_{4p}$. Because the permutation operations are just simple reordering of the data, the circuit size of this block is negligible. TE array block performs the 1D 4-point DCT $\tilde{C}_{4p}\,Z_{4p}\,\tilde{C}_{4p}^{\mathrm{T}}=Y_{4p}^{'}$ in (11). It consists of four TE_sub blocks (TE_sub0 ~ TE_sub3). Each TE_sub block processes data at a rate of four pixels per cycle in parallel and hence TE array block can transform at a rate of 16 pixels per cycle. The adders in this block are shared when performing (17). Post-reorder block accepts 16 pixels per cycle from TE array block and stores them in Buffer block. It also reorders the data by performing the permutation operations $Y_{4p} = P_{4c} Y'_{4p} P_{4c}$ in (11). Buffer block stores the 64 intermediate results, $\tilde{C}_{00}Z_{00}$, $\tilde{C}_{10}Z_{00}$, $\tilde{C}_{00}Z_{01}$, $\tilde{C}_{10}Z_{01}$ that will be added to another 64 intermediate results, $\tilde{C}_{01}Z_{10}$, $\tilde{C}_{11}Z_{10}$, $\tilde{C}_{01}Z_{11}$, $\tilde{C}_{11}Z_{11}$ in order to perform (17). It also plays a role of transpose buffers, i.e., stores the first 1D DCT restuls, transposes them and then provides the transposed data for the second 1D DCT. Its another role is to reorder the data before sending 2D DCT results to outside. Buffer block consists of four 4x48-bit dual-port SRAM's. Controller block selects either 4-point DCT or 8-point DCT to be performed. It also controls the SRAM's in Buffer block.

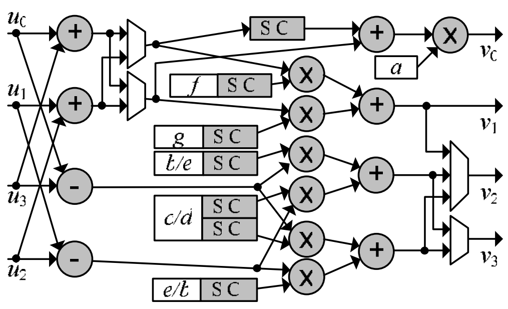


Figure 5. Architecture of proposed transform array element.

Fig. 5 illustrates the architecture of the transform array element. There are four transform array elements in TE array block. One transform array element has eight adders, seven multipliers, seven sign converters (S.C.) and four multiplexers. In order to use the same circuit for 4-point and 8-point DCT's, seven coefficient values can be loaded into the element from outside. In case of 8-point DCT, seven coefficients ($a \sim g$) will be loaded. In case of 4-point DCT, four coefficients ($a = f = g = h$, $b = e = i$, $c = d = j$) will be loaded using $\tilde{C}_{00}$ in (16). Since one transform array element runs at a rate of four pixels per cycle, we need four transform array elements as shown in Fig. 4 to have the rate of 16 pixels per cycle.

We described the proposed transform circuit at RTL (register transfer level) using Verilog HDL (hardware description language) and verified its operation using Xilinx Virtex4 LX60 FPGA (field programmable gate array). The 8-point DCT was applied to the MPEG-4 CODEC circuit, and the 4-point DCT was applied to the H.264 CODEC circuit. The RTL circuit was synthesized into gate-level circuit using 130nm standard cells. The number of logic gates is 30,173 and the maximum operating frequency is 116.8MHz. The required number of clock cycles per one macroblock is 180 for 8-point DCT and 104 for 4-point DCT as shown Table 1. Our circuit uses four 4x48-bit dual-port SRAM's, a total of 96 bytes. We can see that the proposed circuit shows better performance and smaller memory size compared to [5,7,8].

4. Conclusions

We proposed the architecture of transform circuit that can support a variety of transforms based on DCT. The transform coefficients can be loaded from external sources according to the transform type. A small memory is used to store intermediate results, to transpose the first 1D transform results and to reorder the output data. The proposed transform circuit can be used to support multi-standard video CODEC for the multimedia systems such as portable multimedia player (PMP) and digital multimedia broadcasting (DMB). It is also expected to be used in any next-generation transform based on DCT.

Table 1. Performance Comparison

	Cycles/MB (8x8/4x4)	# of gates	Memory (Bytes)	Can support:			
				J	M	H	V
[5]	768/831	32K	D: 128	o	o	o	o
[7]	N.A./1063	50K	D: 384, S: 512	o	o	o	x
[8]	1280/1280	N.A.	N.A.	x	o	o	o
Ours	180/104	30K	D: 96	o	o	o	o

J: JPEG, M: MPEG-1/2/4, H: H.264, V: VC-1
D: dual-port SRAM, S: single-port SRAM

Acknowledgement

This work was sponsored by ETRI SoC Industry Promotion Center, Human Resource Development Project for IT SoC Architect and the CAD tools of IDEC.

References

[1] *CCITT Recommendation T.81, Digital Compression and Coding Continuous-Tone Still Images*, 1992.

[2] *ISO/IEC 14496-2, Coding of Audio-Visual Objects – part2: Visual*, Nov. 1997.

[3] *Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264/ISO/IEC 14496-10 AVC)*, Mar. 2003.

[4] *SMPTE, Standards for Television: VC-1 Compressed Video Bitstream Format and Decoding Process*, SMPTE 421M-2006.

[5] S. Lee and K. Cho, "Architecture of Transform Circuit for Video Decoder Supporting Muliple Standards," *IET Electronics Letters*, vol.44, no.4, pp.274-275, Feb. 2008.

[6] C. P. Fan, "Fast 2-Dimensional 4x4 Forward Integer Transform Implementation for H.264/AVC," *IEEE Transactions on Circuits and Systems II*, vol.53, pp.174-177, Mar. 2006.

[7] J. H. Park, S. H. Lee, K. S. Lim, J. H. Kim and S. Kim, "A Flexible Transform Processor Architecture for multi-CODECs (JPEG, MPEG-2, 4 and H.264)," *IEEE International Symposium on Circuits and Systems*, pp.5347–5350, May 2006.

[8] M. Hase, K. Akie, M. Nobori and K. Matsumoto, "Development of Low-power and Real-time VC-1/H.264/ MPEG-4 Video Processing Hardware," *Asia and South Pacific Design Automation Conference*, pp.637–643, Jan. 2007.