Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2010, Article ID 185398, 21 pages
doi:10.1155/2010/185398

Research Article

An Efficient VLSI Linear Array for DCT/IDCT Using Subband Decomposition Algorithm

Tze-Yun Sung,[1] Yaw-Shih Shieh,[1] and Hsi-Chin Hsin[2]

[1] *Department of Microelectronics Engineering, Chung Hua University, Hsinchu City 300-12, Taiwan*
[2] *Department of Computer Science and Information Engineering, National United University, Miaoli 360-03, Taiwan*

Correspondence should be addressed to Tze-Yun Sung, bobsung@chu.edu.tw

Received 30 January 2010; Accepted 22 March 2010

Academic Editor: Ming Li

Discrete Cosine transform (DCT) and inverse DCT (IDCT) have been widely used in many image processing systems and real-time computation of nonlinear time series. In this paper, a novel lineararray of DCT and IDCT is derived from the data flow of subband decompositions representing the factorized coefficient matrices in the matrix formulation of the recursive algorithm. For increasing the throughput as well as decreasing the hardware cost, the input and output data are reordered. The proposed 8-point DCT/IDCT processor with four multipliers, simple adders, and less registers and ROM storing the immediate results and coefficients, respectively, has been implemented on FPGA (field programmable gate array) and SoC (system on chip). The linear-array DCT/IDCT processor with the computation complexity $O(5N/8)$ and hardware complexity $O(5N/8)$ is fully pipelined and scalable for variable-length DCT/IDCT computations.

1. Introduction

With rapid growth of modern communication applications and computer technologies, image compression and real-time computation of nonlinear time series continues to be in great demand. Discrete Cosine transform (DCT) is one of the major operations in various image/video compression standards [1] and nonlinear time series applications [2–8]. Though fast Fourier transform (FFT) can be used to implement DCT, it requires complex-valued computations; and moreover, N-point DCT by FFT contains $O(\log 2N + 1)$ stages. The conventional DCT architectures using distributed arithmetic involve complex hardware with a great number of registers [9–19]. Other commonly used DCT architectures with matrix formulation and distributed memory [20–27] are however not suited for VLSI

implementation because the hardware complex is proportional to the length of DCT, which leads to the scalability problem of variable-length DCT computations. In this paper, we propose the novel linear-array architecture for scalable DCT/IDCT implementation.

The remainder of this paper proceeds as follows. In Section 2, we propose the fast DCT/IDCT computation based on subband decomposition algorithm. In Section 3, the reconfigurable FPGA-based and programmable SoC implementations with low hardware cost are proposed for the fast DCT/IDCT computation. The performance comparison with conclusions can be found in Section 4.

2. Proposed Fast DCT/IDCT Computation

For an N-point signal, $x[n]$, the discrete cosine transform (DCT) [28] is defined as

$$C[k] = \alpha[k] \sum_{n=0}^{N-1} x[n] \cos\left[\frac{(2n+1)k\pi}{2N}\right], \tag{2.1}$$

where $k = 0, \ldots, N-1$, $\alpha[0] = 1/\sqrt{N}$, and $\alpha[k] = \sqrt{2/N}$ for $k > 0$. Let $x_L[n]$ and $x_H[n]$ denote the low-frequency and high-frequency subband signals of $x[n]$, respectively, which are defined as

$$\begin{aligned} x_L[n] &= \frac{1}{2}\{x[2n] + x[2n+1]\}, \\ x_H[n] &= \frac{1}{2}\{x[2n] - x[2n+1]\}, \end{aligned} \tag{2.2}$$

where $n = 0, 1, 2, \ldots, (N/2)-1$. The original signal $x[n]$ can be obtained from $x_L[n]$ and $x_H[n]$ as follows:

$$\begin{aligned} x[2n] &= x_L[n] + x_H[n], \\ x[2n+1] &= x_L[n] - x_H[n]. \end{aligned} \tag{2.3}$$

As one can see, the DCT of $x[n]$ can be rewritten as

$$\begin{aligned} C[k] &= \sum_{n=0}^{(N/2)-1} \alpha[k]x[2n] \cos\left(\frac{(4n+1)k\pi}{2N}\right) + \sum_{n=0}^{(N/2)-1} \alpha[k]x[2n+1] \cos\left(\frac{(4n+3)k\pi}{2N}\right) \\ &= \underbrace{2\cos\left(\frac{\pi k}{2N}\right) \sum_{n=0}^{(N/2)-1} \alpha[k]x_L[n] \cos\left(\frac{(2n+1)k\pi}{N}\right)}_{C_L[k]} \\ &\quad + \underbrace{2\sin\left(\frac{\pi k}{2N}\right) \sum_{n=0}^{(N/2)-1} \alpha[k]x_H[n] \sin\left(\frac{(2n+1)k\pi}{N}\right)}_{S_H[k]}, \end{aligned} \tag{2.4}$$

where $C_L[k]$ and $S_H[k]$ are the subband DCT and DST (discrete sine transform) of $x[n]$, respectively.

2.1. Fast DCT Computation Based on Subband Decomposition Algorithm

Without loss of generality, the 8-point fast DCT based on subband decomposition algorithm is proposed for the widely used JPEG and MPEG-1/2 standards, which can be easily extended to variable-length DCT computations. The vector form of 8-point DCT can be written as

$$\mathbf{C}_8 = \left[\mathbf{T}_{\text{SB\_DCT},8} \;\; \mathbf{T}_{\text{SB\_DST},8}\right]_{8\times 8} \cdot \begin{bmatrix} \mathbf{x}_L \\ \mathbf{x}_H \end{bmatrix}_{8\times 1}, \tag{2.5}$$

where $\mathbf{C}_8 = [C[0]\cdots C[7]]^T$, $\mathbf{x}_L = [x_L[0]\cdots x_L[3]]^T$, $\mathbf{x}_H = [x_H[0]\cdots x_H[3]]^T$, and $\mathbf{T}_{SB\_DCT,8}$ and $\mathbf{T}_{SB\_DST,8}$ denote the 8×4 matrices of subband DCT and subband DST, respectively, which can form orthonormal bases for the two orthogonal subspaces of $\mathbf{R}^8$. Notice that, due to the orthogonality between $\mathbf{T}_{SB\_DCT,8}$ and $\mathbf{T}_{SB\_DST,8}$, $x_L[n]$ and $x_H[n]$ can be obtained from $C[k]$ as follows:

$$\begin{aligned} x_L[n] &= \sum_{n=0}^{N-1} \alpha[k] \cos\left(\frac{\pi k}{2N}\right) C[k] \cos\left(\frac{(2n+1)k\pi}{N}\right), \\ x_H[n] &= \sum_{n=0}^{N-1} \alpha[k] \sin\left(\frac{\pi k}{2N}\right) C[k] \sin\left(\frac{(2n+1)k\pi}{N}\right), \end{aligned} \tag{2.6}$$

where $n = 0, 1, 2, \ldots, N/2-1$, and $N = 8$.

The proposed fast DCT algorithm is a subband decomposition-based multistage algorithm. Specifically, let

$$\begin{aligned} x_{LL}[n] &= \frac{1}{2}\{x_L[2n] + x_L[2n+1]\}, \\ x_{LH}[n] &= \frac{1}{2}\{x_L[2n] - x_L[2n+1]\}, \\ x_{HL}[n] &= \frac{1}{2}\{x_H[2n] + x_H[2n+1]\}, \\ x_{HH}[n] &= \frac{1}{2}\{x_H[2n] - x_H[2n+1]\}, \end{aligned} \tag{2.7}$$

$$\mathbf{M}_8 = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 \\ 0.5 & -0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & -0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & -0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & -0.5 \end{bmatrix}$$

$$\mathbf{M}_4 = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 \\ 0.5 & -0.5 & 0 & 0 \\ 0 & 0 & 0.5 & -0.5 \end{bmatrix}$$

$$\mathbf{M}_2 = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & -0.5 \end{bmatrix}$$

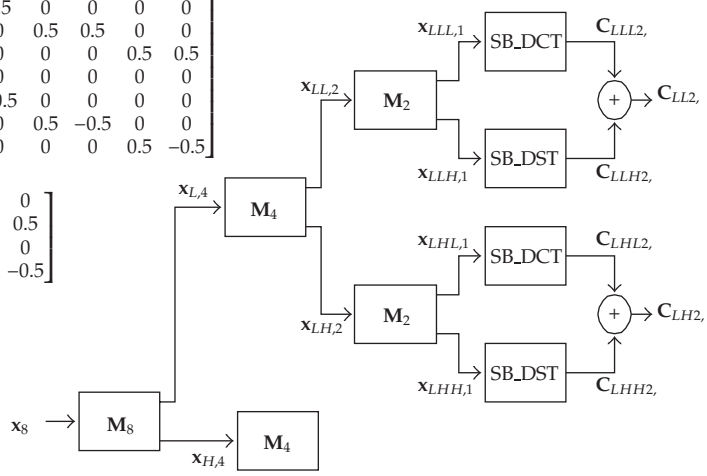


Figure 1: Data flow of computing the 2-point subband DCT: $\mathbf{C}_{LL,2}$ and subband DST: $\mathbf{C}_{LH,2}$ (for the 8-point DCT of the input signal: $\mathbf{x}_8$) based on subband decomposition.

where $n = 0, 1$. And let

$$\begin{aligned} x_{LLL}[n] &= \frac{1}{2}\{x_{LL}[2n] + x_{LL}[2n+1]\}, \\ x_{LLH}[n] &= \frac{1}{2}\{x_{LL}[2n] - x_{LL}[2n+1]\}, \\ x_{LHL}[n] &= \frac{1}{2}\{x_{LH}[2n] + x_{LH}[2n+1]\}, \\ x_{LHH}[n] &= \frac{1}{2}\{x_{LH}[2n] - x_{LH}[2n+1]\}, \\ x_{HLL}[n] &= \frac{1}{2}\{x_{HL}[2n] + x_{HL}[2n+1]\}, \\ x_{HLH}[n] &= \frac{1}{2}\{x_{HL}[2n] - x_{HL}[2n+1]\}, \\ x_{HHL}[n] &= \frac{1}{2}\{x_{HH}[2n] + x_{HH}[2n+1]\}, \\ x_{HHH}[n] &= \frac{1}{2}\{x_{HH}[2n] - x_{HH}[2n+1]\}, \end{aligned} \tag{2.8}$$

where $n = 0$. Based on subband decompositions using (2.2), (2.7), and (2.8), data flow of computing the 2-point subband DCT: $\mathbf{C}_{LL,2}$ and subband DST: $\mathbf{C}_{LH,2}$ for the 8-point DCT is shown in Figure 1. As one can see, data flow of computing $\mathbf{C}_{HL,2}$ and $\mathbf{C}_{HH,2}$ can be obtained

in a similar way, and therefore is not shown in Figure 1. All of the 2-point subband DCTs and DSTs are given by

$$\begin{aligned}
\mathbf{C}_{LL,2} &= \left[\mathbf{T}_{\text{SB_DCT},2} \;\; \mathbf{T}_{\text{SB_DST},2}\right]_{2\times 2} \cdot \begin{bmatrix}\mathbf{x}_{LLL}\\ \mathbf{x}_{LLH}\end{bmatrix}_{2\times 1} = \underbrace{\mathbf{T}_{\text{SB_DCT},2}\cdot\mathbf{x}_{LLL}}_{\hat{C}_{LLL,2}} + \underbrace{\mathbf{T}_{\text{SB_DST},2}\cdot\mathbf{x}_{LLH}}_{\hat{S}_{LLH,2}},\\
\mathbf{C}_{LH,2} &= \left[\mathbf{T}_{\text{SB_DCT},2} \;\; \mathbf{T}_{\text{SB_DST},2}\right]_{2\times 2} \cdot \begin{bmatrix}\mathbf{x}_{LHL}\\ \mathbf{x}_{LHH}\end{bmatrix}_{2\times 1} = \underbrace{\mathbf{T}_{\text{SB_DCT},2}\cdot\mathbf{x}_{LHL}}_{\hat{C}_{LHL,2}} + \underbrace{\mathbf{T}_{\text{SB_DST},2}\cdot\mathbf{x}_{LHH}}_{\hat{S}_{LHH,2}},\\
\mathbf{C}_{HL,2} &= \left[\mathbf{T}_{\text{SB_DCT},2} \;\; \mathbf{T}_{\text{SB_DST},2}\right]_{2\times 2} \cdot \begin{bmatrix}\mathbf{x}_{HLL}\\ \mathbf{x}_{HLH}\end{bmatrix}_{2\times 1} = \underbrace{\mathbf{T}_{\text{SB_DCT},2}\cdot\mathbf{x}_{HLL}}_{\hat{C}_{HLL,2}} + \underbrace{\mathbf{T}_{\text{SB_DST},2}\cdot\mathbf{x}_{HLH}}_{\hat{S}_{HLH,2}},\\
\mathbf{C}_{HH,2} &= \left[\mathbf{T}_{\text{SB_DCT},2} \;\; \mathbf{T}_{\text{SB_DST},2}\right]_{2\times 2} \cdot \begin{bmatrix}\mathbf{x}_{HHL}\\ \mathbf{x}_{HHH}\end{bmatrix}_{2\times 1} = \underbrace{\mathbf{T}_{\text{SB_DCT},2}\cdot\mathbf{x}_{HHL}}_{\hat{C}_{HHL,2}} + \underbrace{\mathbf{T}_{\text{SB_DST},2}\cdot\mathbf{x}_{HHH}}_{\hat{S}_{HHH,2}}.
\end{aligned} \tag{2.9}$$

Thus, we have

$$\begin{bmatrix}\mathbf{C}_{LL,2}\\ \mathbf{C}_{LH,2}\\ \mathbf{C}_{HL,2}\\ \mathbf{C}_{HH,2}\end{bmatrix} = \mathbf{R}_8 \cdot \mathbf{x}_8, \tag{2.10}$$

where $\mathbf{x}_8 = [x[0]\cdots x[7]]^T$ is the original signal, and

$$\mathbf{R}_8 = \frac{\sqrt{2}}{8}\cdot\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1\\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1\\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1\\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1\\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1\\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1\\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1
\end{bmatrix}. \tag{2.11}$$

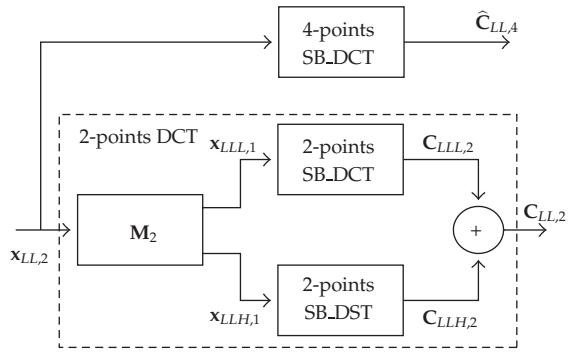


Figure 2: Data flow of computing $\widehat{\mathbf{C}}_{LL,4}$ and $\mathbf{C}_{LL,2}$ based on subband decomposition.

Similarly, we have the following:

$$\begin{aligned}
\mathbf{C}_{L,4} &= \left[\mathbf{T}_{\text{SB_DCT},4} \quad \mathbf{T}_{\text{SB_DST},4}\right]_{4\times4} \cdot \begin{bmatrix}\mathbf{x}_{LL,2}\\ \mathbf{x}_{LH,2}\end{bmatrix}_{4\times1} = \underbrace{\mathbf{T}_{\text{SB_DCT},4}\cdot\mathbf{x}_{LL,2}}_{\widehat{C}_{LL,4}} + \underbrace{\mathbf{T}_{\text{SB_DST},4}\cdot\mathbf{x}_{LH,2}}_{\widehat{S}_{LH,4}},\\
\mathbf{C}_{H,4} &= \left[\mathbf{T}_{\text{SB_DCT},4} \quad \mathbf{T}_{\text{SB_DST},4}\right]_{4\times4} \cdot \begin{bmatrix}\mathbf{x}_{HL,2}\\ \mathbf{x}_{HH,2}\end{bmatrix}_{4\times1} = \underbrace{\mathbf{T}_{\text{SB_DCT},4}\cdot\mathbf{x}_{HL,2}}_{\widehat{C}_{HL,4}} + \underbrace{\mathbf{T}_{\text{SB_DST},4}\cdot\mathbf{x}_{HH,2}}_{\widehat{S}_{HH,4}}.
\end{aligned} \tag{2.12}$$

Figure 2 depicts the relationship between $\widehat{\mathbf{C}}_{LL,4}$ and $\mathbf{C}_{LL,2}$, which can be obtained by the following:

$$\widehat{\mathbf{C}}_{LL,4} = \mathbf{T}_{SB\_DCT,4}\cdot\mathbf{x}_{LL,2}, \tag{2.13}$$

$$\mathbf{C}_{LL,2} = \mathbf{T}_2\cdot x_{LL,2}, \tag{2.14}$$

where $\mathbf{T}_2$ is the 2×2 transform matrix of the conventional 2-point DCT. Hence, (2.13) can be rewritten as

$$\widehat{\mathbf{C}}_{LL,4} = \mathbf{T}_{SB\_DCT,4}\cdot\mathbf{T}_2^{-1}\cdot\mathbf{C}_{LL,2} = \begin{bmatrix}1.4142 & 0\\ 0 & 1.3066\\ 0 & 0\\ 0 & -0.5412\end{bmatrix}\cdot\mathbf{C}_{LL,2}. \tag{2.15}$$

The relationship between $\widehat{\mathbf{S}}_{LH,4}$ and $\mathbf{C}_{LH,2}$ shown in Figure 3 is based on the following:

$$\begin{aligned}
\widehat{\mathbf{S}}_{LH,4} &= \mathbf{T}_{SB_DST,4}\cdot x_{LH,2},\\
\mathbf{C}_{LH,2} &= \mathbf{T}_2\cdot x_{LH,2}.
\end{aligned} \tag{2.16}$$

Thus, we have

$$\widehat{\mathbf{S}}_{LH,4} = \mathbf{T}_{SB\_DST,4} \cdot \mathbf{T}_2^{-1} \cdot \mathbf{C}_{LH,2} = \begin{bmatrix} 0 & 0 \\ 0.5412 & 0 \\ 0 & 1.4142 \\ 1.3066 & 0 \end{bmatrix} \cdot \mathbf{C}_{LH,2}. \tag{2.17}$$

Similarly, based on (2.5) and the following equations:

$$\begin{aligned} \mathbf{C}_{L,4} &= \mathbf{T}_4 \cdot \mathbf{x}_{L,4}, \\ \mathbf{C}_{H,4} &= \mathbf{T}_4 \cdot x_{H,4}, \end{aligned} \tag{2.18}$$

where $\mathbf{T}_4$ is the 4×4 transform matrix of the conventional 4-point DCTs, we have

$$\begin{aligned} \widehat{\mathbf{C}}_{L,8} &= \mathbf{T}_{SB\_DCT,8} \cdot x_{L,4} \\ &= \mathbf{T}_{SB\_DCT,8} \cdot \mathbf{T}_4^{-1} \cdot \mathbf{C}_{L,4} \\ &= \begin{bmatrix} 1.412 & 0 & 0 & 0 \\ 0 & 1.3870 & 0 & 0 \\ 0 & 0 & 1.3066 & 0 \\ 0 & 0 & 0 & 1.1759 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.7857 \\ 0 & 0 & -0.5412 & 0 \\ 0 & -0.2759 & 0 & 0 \end{bmatrix} \cdot \mathbf{C}_{L,4}, \end{aligned} \tag{2.19}$$

$$\begin{aligned} \widehat{\mathbf{C}}_{H,8} &= \mathbf{T}_{SB\_DST,8} \cdot \mathbf{x}_{H,4} \\ &= \mathbf{T}_{SB\_DST,8} \cdot \mathbf{T}_4^{-1} \cdot \mathbf{C}_{H,4} \\ &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.2549 & 0 & -0.1056 & 0 \\ 0 & 0.5 & 0 & -0.2071 \\ 0.3007 & 0 & 0.7259 & 0 \\ 0 & 0.5412 & 0 & 1.3066 \\ 0.4500 & 0 & 1.0864 & 0 \\ 0 & 1.2071 & 0 & -0.5 \\ 1.2815 & 0 & -0.5308 & 0 \end{bmatrix} \cdot \mathbf{C}_{H,4}. \end{aligned} \tag{2.20}$$

Figure 4 depicts data flow of computing $\mathbf{C}_{L,4}$ and $\mathbf{C}_{H,4}$ using 4-point subband DCT and DST. Figure 5 depicts data flow of computing $\widehat{\mathbf{C}}_{L,8}$ and $\mathbf{C}_{L,4}$ based on subband decomposition. Data flow of computing $\widehat{\mathbf{S}}_{H,8}$ and $\mathbf{C}_{H,4}$ based on subband decomposition is shown in Figure 6. Data flow of computing $\mathbf{C}_8$ using 8-point subband DCT and DST is shown in Figure 7. In other words, $\mathbf{C}_8$ can be obtained by

$$\mathbf{C}_8 = \widehat{\mathbf{C}}_{L,8} + \widehat{\mathbf{S}}_{H,8}. \tag{2.21}$$

Base on (2.12), (2.15), (2.17), (2.19) and (2.20), we have

$$\mathbf{C}_8 = \mathbf{F}_8 \cdot \left[\mathbf{C}_{LL,2}^T \;\; \mathbf{C}_{LH,2}^T \;\; \mathbf{C}_{HL,2}^T \;\; \mathbf{C}_{HH,2}^T\right]^T, \tag{2.22}$$

where

$$\mathbf{F}_8 = \left[\mathbf{K}_3 \;\; \mathbf{K}_4\right]_{8\times 8} \cdot \begin{bmatrix} \left[\mathbf{K}_1 \;\; \mathbf{K}_2\right]_{4\times 4} & 0 \\ 0 & \left[\mathbf{K}_1 \;\; \mathbf{K}_2\right]_{4\times 4} \end{bmatrix}_{8\times 8}, \tag{2.23}$$

$$\mathbf{K}_1 = \begin{bmatrix} 1.4142 & 0 \\ 0 & 1.3066 \\ 0 & 0 \\ 0 & -0.5412 \end{bmatrix}, \tag{2.24}$$

$$\mathbf{K}_2 = \begin{bmatrix} 0 & 0 \\ 0.5412 & 0 \\ 0 & 1.4142 \\ 1.3066 & 0 \end{bmatrix}, \tag{2.25}$$

$$\mathbf{K}_3 = \begin{bmatrix} 1.412 & 0 & 0 & 0 \\ 0 & 1.3870 & 0 & 0 \\ 0 & 0 & 1.3066 & 0 \\ 0 & 0 & 0 & 1.1759 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.7857 \\ 0 & 0 & -0.5412 & 0 \\ 0 & -0.2759 & 0 & 0 \end{bmatrix}, \tag{2.26}$$

$$\mathbf{K}_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.2549 & 0 & -0.1056 & 0 \\ 0 & 0.5 & 0 & -0.2071 \\ 0.3007 & 0 & 0.7259 & 0 \\ 0 & 0.5412 & 0 & 1.3066 \\ 0.4500 & 0 & 1.0864 & 0 \\ 0 & 1.2071 & 0 & -0.5 \\ 1.2815 & 0 & -0.5308 & 0 \end{bmatrix}. \tag{2.27}$$

According to (2.24)–(2.27), we have

$$\mathbf{F}_8 = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.8123 & 0.7507 & 0 & 0.3605 & 0 & 0 & -0.1493 \\ 0 & 0 & 0 & 1.8478 & 0 & 0.7654 & 0 & 0 \\ 0 & -0.6364 & 1.5364 & 0 & 0.4252 & 0 & 0 & 1.0266 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0.4252 & -1.0266 & 0 & 0.6364 & 0 & 0 & 1.5364 \\ 0 & 0 & 0 & -0.7654 & 0 & 1.8478 & 0 & 0 \\ 0 & -0.3605 & -0.1493 & 0 & 1.8123 & 0 & 0 & -0.7507 \end{bmatrix}. \tag{2.28}$$

Finally, the proposed 8-point DCT computation based on subband decomposition is as follows:

$$\mathbf{C}_8 = \widehat{\mathbf{F}}_8 \cdot \mathbf{R}_8 \cdot \mathbf{x}_8, \tag{2.29}$$

where

$$\widehat{\mathbf{F}}_8 = 2 \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.9239 & 0.3827 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.3827 & 0.9239 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.9062 & 0.3754 & 0.1802 & -0.0746 \\ 0 & 0 & 0 & 0 & -0.1802 & -0.0746 & 0.9062 & -0.3754 \\ 0 & 0 & 0 & 0 & -0.3182 & 0.7682 & 0.2126 & 0.5133 \\ 0 & 0 & 0 & 0 & 0.2126 & -0.5133 & 0.3182 & 0.7682 \end{bmatrix}. \tag{2.30}$$

Figure 8 shows block diagram of the proposed DCT computation; one of the advantages is that $\mathbf{R}_8$ is orthogonal, and all of the submatrices of $\widehat{\mathbf{F}}_8$ are orthonormal.

2.2. Fast IDCT Computation Based on Subband Decomposition Algorithm

According to (2.29), IDCT can be obtained by

$$\mathbf{x}_8 = \mathbf{R}_8^{-1} \cdot \widehat{\mathbf{F}}_8^{-1} \cdot \mathbf{C}_8, \tag{2.31}$$

where

$$\begin{aligned}
\mathbf{R}_8^{-1} &= \frac{8}{\sqrt{2}} \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1
\end{bmatrix}, \\
\widehat{\mathbf{F}}_8^{-1} &= \frac{1}{2} \cdot \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.9239 & -0.3827 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.3827 & 0.9239 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.9062 & -0.1802 & -0.3182 & 0.2126 \\
0 & 0 & 0 & 0 & 0.3754 & 0.3754 & 0.7682 & -0.5133 \\
0 & 0 & 0 & 0 & 0.1802 & 0.1802 & 0.2126 & 0.3182 \\
0 & 0 & 0 & 0 & -0.0746 & -0.0746 & 0.5133 & 0.7682
\end{bmatrix}.
\end{aligned} \tag{2.32}$$

As $\mathbf{R}_8$ is orthogonal and all of the submatrices of $\widehat{\mathbf{F}}_8$ are orthonormal, the inverse of $\mathbf{R}_8$ and $\widehat{\mathbf{F}}_8$ can be obtained easily. In addition, it takes only twenty multiplication operations for both DCT and IDCT.

3. VLSI Implementation of an Efficient Linear-Array DCT/IDCT Processor

Based on the proposed approach to fast DCT computation shown in Figure 8, an efficient architecture for implementing the fast DCT/IDCT processor is thus presented in this section. Recall that the DCT of a signal, $\mathbf{x}_8$, can be efficiently obtained by $\mathbf{C}_8 = \widehat{\mathbf{F}}_8 \cdot \mathbf{R}_8 \cdot \mathbf{x}_8$. Let $\mathbf{y}_8 = \mathbf{R}_8 \cdot \mathbf{x}_8$, then we have $\mathbf{C}_8 = \widehat{\mathbf{F}}_8 \cdot \mathbf{y}_8$. Figure 9 shows the matrix-vector multiplication of $\mathbf{R}_8 \cdot \mathbf{x}_8$, in which six CSA(3,2)s (carry-save-adder (3,2)) and one CSA (carry-save-adder) [29, 30] are

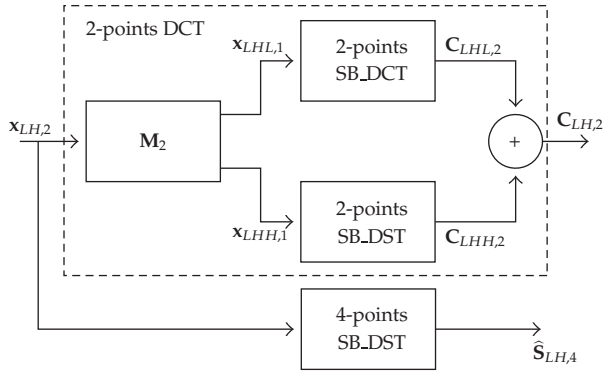


Figure 3: Data flow of computing $\mathbf{C}_{LH,2}$ and $\widehat{\mathbf{S}}_{LH,4}$ based on subband decomposition.

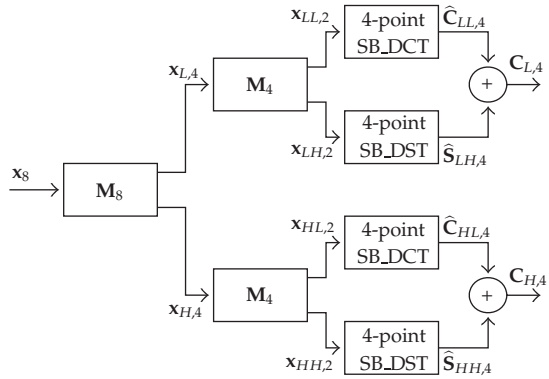


Figure 4: Data flow of computing $\mathbf{C}_{L,4}$ and $\mathbf{C}_{H,4}$ using 4-point subband DCT and DST.

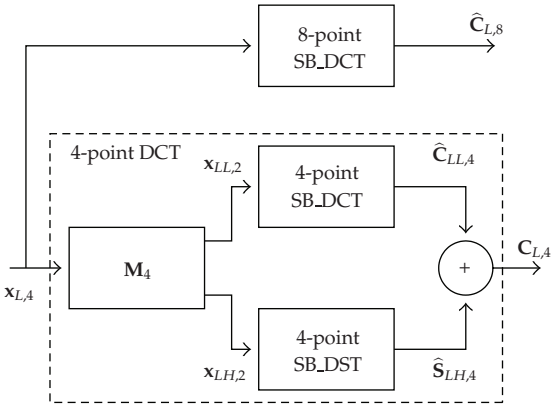


Figure 5: Data flow of computing $\widehat{\mathbf{C}}_{L,8}$ and $\mathbf{C}_{L,4}$ based on subband decomposition.

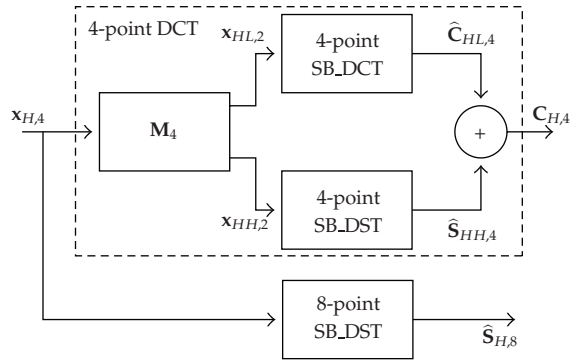


Figure 6: Data flow of computing $\widehat{\mathbf{S}}_{H,8}$ and $\mathbf{C}_{H,4}$ based on subband decomposition.

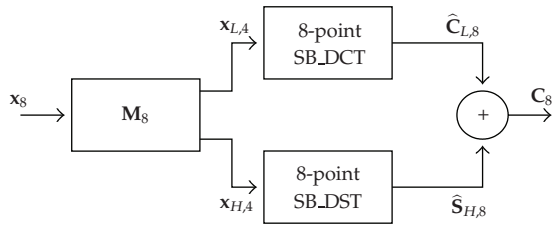
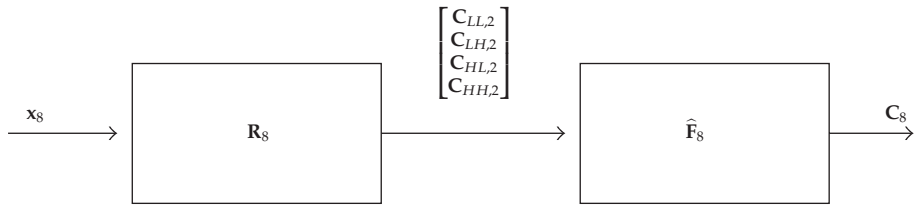


Figure 7: Data flow of computing $\mathbf{C}_8$ using 8-point subband DCT and DST.



$$\mathbf{R}_8 = \frac{\sqrt{2}}{8}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

$$\widehat{\mathbf{F}}_8 = 2\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.9239 & 0.3827 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.3827 & 0.9239 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.9062 & 0.3754 & 0.1802 & -0.0746 \\ 0 & 0 & 0 & 0 & -0.1802 & -0.0746 & 0.9062 & -0.3754 \\ 0 & 0 & 0 & 0 & -0.3182 & 0.7682 & 0.2126 & 0.5133 \\ 0 & 0 & 0 & 0 & 0.2126 & -0.5133 & 0.3182 & 0.7682 \end{bmatrix}$$

Figure 8: Block diagram of the proposed (8-point) fast DCT algorithm based on subband decomposition.

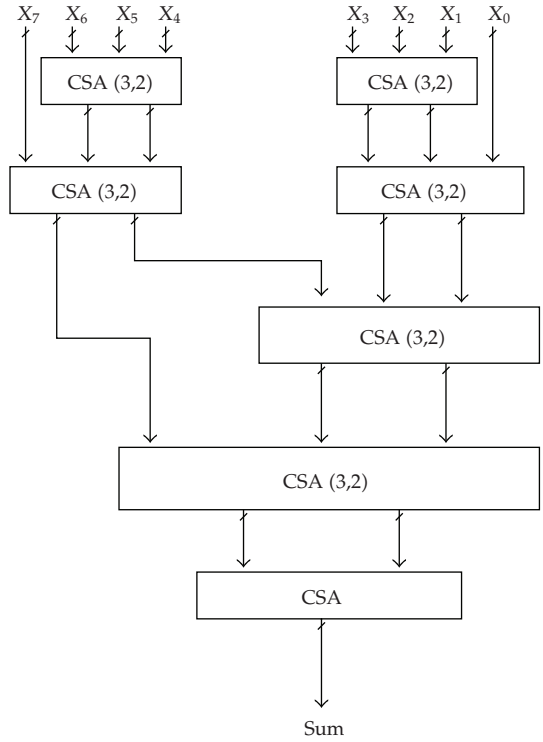


Figure 9: Fast adder (FA) for the matrix-vector multiplication of $\mathbf{R}_8 \cdot \mathbf{x}_8$. (Note: The width of buses is 32-bit.)

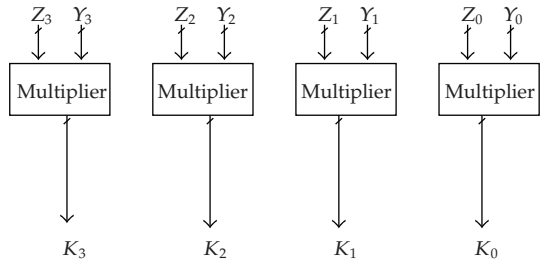


Figure 10: Multiplier array (MA) consisted of four multipliers. (Note: The width of buses is 32-bit.)

utilized, and therefore four simple-addition time and one CSA computation time is required to compute each element of $\mathbf{y}_8$. Figures 10 and 11 show the Multiplier array (MA) consisted of four multipliers and the CSA array (CA) consisted of eight CSAs, respectively, which are used to compute the matrix-vector computation of $\hat{\mathbf{F}}_8 \cdot \mathbf{y}_8$; thus, only one multiplication time with one CSA computation time is needed to compute each element of $\mathbf{C}_8$, that is, the DCT coefficient. Table 3 depicts data flow of the proposed fast DCT processor with pipelined linear-array architecture [31]. As a result, only five multiplication cycles with five addition cycles are needed to compute 8-point DCT. In general, for N-point DCT, the computation time and hardware complexity of the proposed fast DCT processor are $O(5N/8)$ and $O(N/2)$, respectively.

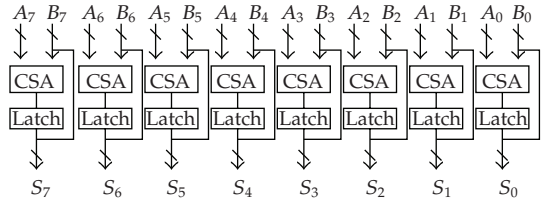


Figure 11: CSA array (CA) consisted of eight CSAs. (Note: The width of buses is 32-bit.)

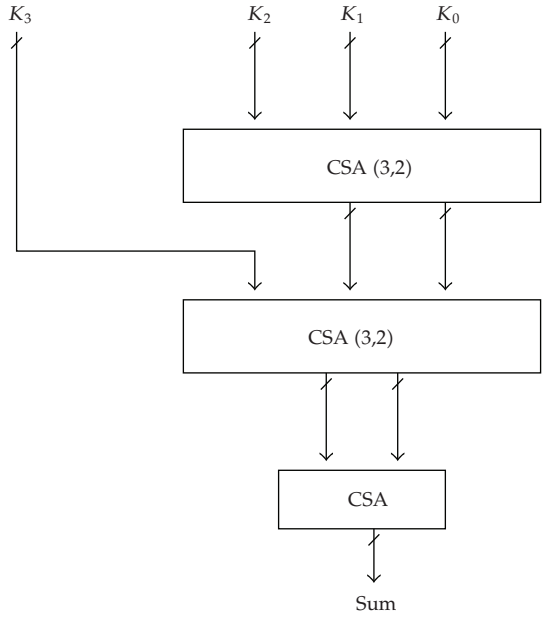


Figure 12: Full CSA(4,2) consisted of two CSA(3,2) and one CSA.

Table 4 shows data flow of the proposed fast IDCT algorithm [31], where $\mathbf{C}_8$ is the DCT of an 8-point signal $\mathbf{x}_8$; $\mathbf{z}_8 = \widehat{\mathbf{F}}_8^{-1} \cdot \mathbf{C}_8$, and $\mathbf{x}_8 = \mathbf{R}_8^{-1} \cdot \mathbf{z}_8$. Figure 12 shows the so-called full CSA(4,2) (FCSA(4,2)) consisted of two CSA(3,2) and one CSA for the computation of $\mathbf{z}_8$ [29, 30]. It is noted that the CSA array consisted of eight CSAs shown in Figure 11 can also be used for the computation of $\mathbf{x}_8$. As shown in Table 4 , only five multiplication cycles with three addition cycles are needed to compute 8-point IDCT. As one can see, the computation time and hardware complexity of the proposed fast IDCT architecture are the same as that of the proposed fast DCT architecture. In addition, only 16-word RAM/registers and 10-word ROM are required to store the intermediate results and constants, respectively; and the latency time is only 5-multiplication-cycle.

Figure 13 shows system block diagram of the proposed fast DCT/IDCT architecture. The platform for architecture development and verification has been designed as well as implemented in order to evaluate the development cost. Figure 14 depicts block diagram of the platform, in which the 8051 microcontroller reads data from PC via DMA channel and writes the result back to PC by USB 2.0 bus; the Xilinx XC2V6000 FPGA chip implements the proposed DCT processor [32]. The architecture development and verification board shown in Figure 15 are to verify and evaluate the proposed DCT/IDCT architecture. Moreover, the

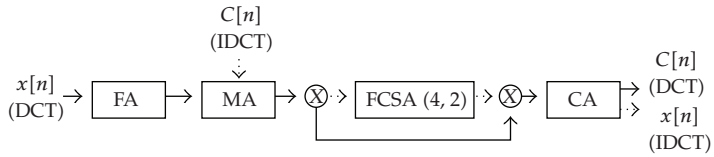


Figure 13: System block diagram of the proposed DCT/IDCT architecture (FA: fast-adder-array, MA: Multiplier array, FCSA(4,2): full CSA(4,2), and CA: CSA- array).

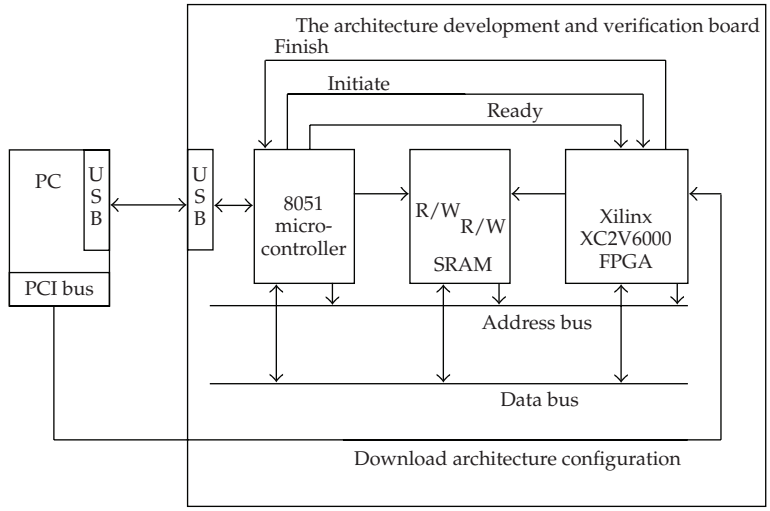


Figure 14: Block diagram of the architecture development and verification platform for the proposed DCT/IDCT processor.

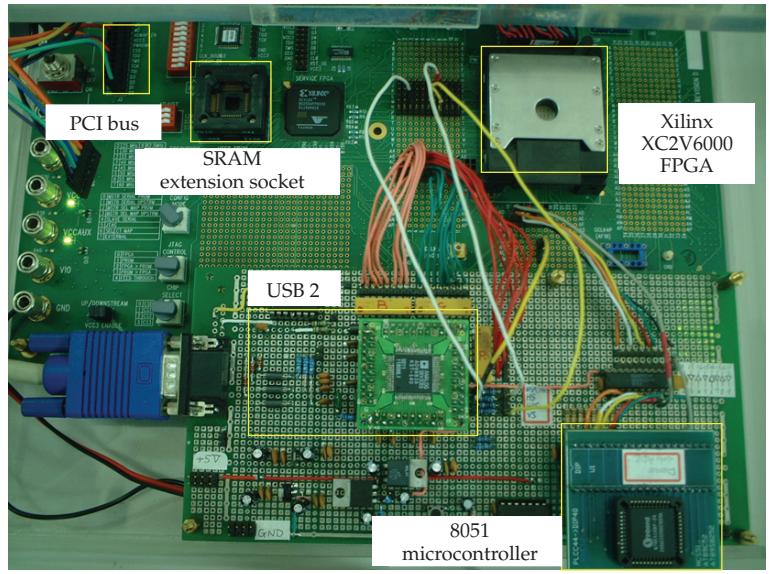


Figure 15: The architecture development and verification board.

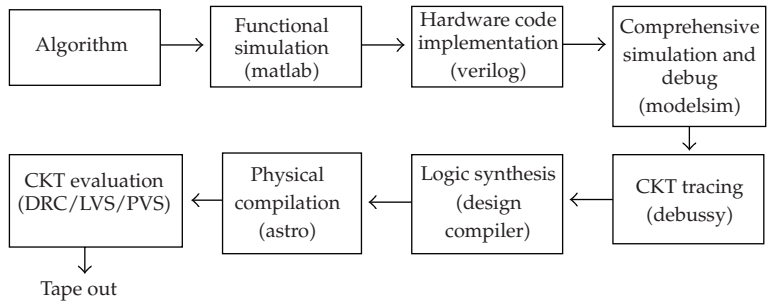


Figure 16: Cell-based design flow.

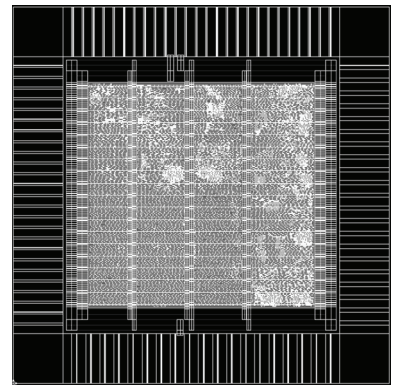


Figure 17: The layout view of the proposed 8-point DCT/IDCT processor with 32-bit operand.

reusable intellectual property (IP) DCT/IDCT core has also been implemented in Matlab for functional simulations. The hardware code written in Verilog is running on a workstation with the ModelSim simulation tool and Xilinx ISE smart compiler. In addition, the FPGA platform shown in Figure 14 is to verify and evaluate the proposed DCT architecture. It is noted that the throughput can be improved by using the proposed architecture while the computation accuracy is the same as that obtained by using the conventional one with the same word length.

The SoC is synthesized by the TSMC 0.18 μm 1P6M CMOS cell libraries [33]. The physical circuit is synthesized by the Astro tool. The circuit is evaluated by DRC, LVS, and PVS [34]. Figure 16 shows the cell-based design flow. The layout view of the 8-point DCT/IDCT processor with 32-bit operand is shown in Figure 17. The core areas are obtained by the Synopsys design analyzer. The power consumptions are obtained by the PrimePower. The reported core size of the implemented the proposed processor is $1520 \times 1520\,\mu\text{m}^2$ and the power dissipation is 102.2 mW at 1.8 V with clock rate of 1 GHz. Thus, the proposed programmable DCT/IDCT architecture is able to improve the power consumption and computation speed significantly. All the control signals are internally generated on-chip. The proposed DCT/IDCT processor provides both high-throughput and low gate count.

The proposed reconfigurable DCT/IDCT processor used to compute 8/16/32/64-point DCT/IDCT on FPGA are composed mainly of the 8-point DCT/IDCT core; the computation complexity using a single 8-point DCT/IDCT core is $O(5N/8)$ for extending

Table 1: Comparisons between the proposed architecture and the conventional architectures.

8-point DCT/IDCT	The conventional architectures		The conventional pipelined architectures	The proposed high-efficient architecture
	The single-processor architectures [9–11]	The parallel architectures with single memory-bank [15–19]	The pipelined architectures with single memory-bank [1, 9–14]	This work(Sung, Shieh and Hsin, 2010)
Processors	1	8	5 (CORDIC)	—
Real multipliers	2	16	0	4
Real adders	3	18	18	26
RAM (Registers)	64	64	64	16
ROM	6	6	6	10
Hardware complexity	$O(1)$	$O(N - \log_2 N + 1)$	$O(N - \log_2 N)$	$O(N/2)$
Computation complexity	$O(N^2)$	$O(2N)$	$O(N)$	$O(5N/8)$
Latency	64	16	8	5
Pipelinability	no	no	yes	yes
Scalability	poor	poor	good	better
Power consumption	poor	poor	good	better

Table 2: Comparisons of the proposed architecture and other commonly used architectures.

8-point	Lee et al. [20]	Chang and Wang [21]	Hsiao and Shiue [22]	Hsiao and Tseng [23]	Hou [24]	Sung [1, 9–14]	This work
DCT/IDCT	DCT/IDCT	DCT/IDCT	DCT	DCT/IDCT	DCT/IDCT	DCT/IDCT	DCT/IDCT
Real multipliers	28	64	—	—	—	—	4
CORDIC processors	—	—	—	—	3	5	—
Real adders	134	88	9	10	14	18	26
Complex multipliers	—	—	3	3		—	—
Delay elements (Words)	256	114	—	171	—	—	—
Memory (Words)	~384	~200	~370	—	—	70	26
Hardware complexity	$O(N \log N)$	$O(N^2)$	$O(\log N)$	$O(\log N)$	$O(\log N)$	$O(N - \log N)$	$O(N/2)$
Computation complexity	$O(\log N)$	$O(N)$	$O(N \log N)$	$O(N \log N)$	$O(N \log N)$	$O(N)$	$O(5N/8)$
Pipelinability	no	no	no	no	yes	yes	yes
Scalability	poor	poor	good	good	good	good	better

Table 3: Data flow of the proposed fast DCT processor with pipelined linear-array architecture (Add.-cycle: addition-cycle and Mul.-cycle: multiplication-cycle).

Processor	FA	MA	CA
Add.-cycle_1	$y[0]$	—	$C[0]$
Add.-cycle_2	$y[1]$	—	$C[1]$
Add.-cycle_3	$y[2]$	—	—
Mul.-cycle_1	$y[3]$	$y[2] \cdot 0.9239$, $y[2] \cdot (-0.3827)$ $y[3] \cdot 0.3827$, $y[3] \cdot (0.9239)$	—
Add.-cycle_4	$y[4]$	—	$C[2]$, $C[3]$
Mul.-cycle_2	$y[5]$	$y[4] \cdot 0.9062$, $y[4] \cdot (-0.1802)$, $y[4] \cdot (-0.3182)$, $y[4] \cdot 0.2126$	—
Mul.-cycle_3	$y[6]$	$y[5] \cdot 0.3754$, $y[5] \cdot (-0.0746)$, $y[5] \cdot 0.7682$, $y[5] \cdot 0.5133$	—
Mul.-cycle_4	$y[7]$	$y[6] \cdot 0.1802$, $y[6] \cdot 0.9062$, $y[6] \cdot 0.2126$, $y[6] \cdot 0.3182$	—
Mul.-cycle_5	—	$y[7] \cdot (-0.0746)$, $y[7] \cdot (-0.3754)$, $y[7] \cdot 0.5133$, $y[7] \cdot 0.7682$	—
Add.-cycle_5	—		$C[4]$, $C[5]$, $C[6]$, $C[7]$

Table 4: Data flow of the proposed fast IDCT processor with pipelined linear-array architecture (Add.-cycle: addition-cycle and Mul.-cycle: multiplication-cycle).

Processor	MA	FCSA(4,2)	CA
Mul.-cycle_1	$C[2] \cdot 0.9239$, $C[3] \cdot (-0.3827)$ $C[2] \cdot 0.3827$, $C[3] \cdot 0.92393$	$z[0]$, $z[1]$	—
Mul.-cycle_2	$C[4] \cdot 0.9062$, $C[5] \cdot (-0.1802)$, $C[6] \cdot (-0.3182)$, $C[7] \cdot 0.2126$	$z[2]$, $z[3]$	C_0 + C_1 = C_01
Mul.-cycle_3	$C[4] \cdot 0.3754$, $C[5] \cdot 0.3754$, $C[6] \cdot 0.7682$, $C[7] \cdot (-0.5133)$	$z[4]$	C_01 + C_2 = C_02
Mul.-cycle_4	$C[4] \cdot (-0.3182)$, $C[5] \cdot 0.7682$, $C[6] \cdot 0.2126$, $C[7] \cdot 0.5144$	$z[5]$	C_02 + C_3 = C_03
Mul.-cycle_5	$C[4] \cdot 0.2126$, $C[5] \cdot (-0.5133)$, $C[6] \cdot 0.3182$, $C[7] \cdot 0.7682$	$z[6]$	C_03 + C_4 = C_04
Add.-cycle_1	—	$z[7]$	C_04 + C_5 = C_05
Add.-cycle_2	—	—	C_05 + C_6 = C_06
Add.-cycle_3	—	—	C_06 + C_7 = C_07 $x[0]$, $x[1]$, $x[2]$, $x[3]$, $x[4]$, $x[5]$, $x[6]$, $x[7]$

N-point DCT/IDCT computation. Note that the transform matrices used for the proposed linear array with 8-point DCT core can be extended to a variety of different sizes. Thus, the proposed architecture is highly scalable.

The linear-array architecture with use of hardware resources has been proposed for trade offs of performance, chip area and power consumption. As a result, it has the advantage of balancing the need for power saving with computation speed.

4. Conclusion

By taking advantage of subband decomposition, a high-efficiency architecture with pipelined structures is proposed for fast DCT/IDCT computation. Specifically, the proposed DCT/IDCT architecture not only improves throughput by more than two times that of the conventional architectures [9–11, 15–19], but also saves memory space significantly [1, 9–22]. Table 1 shows comparisons between the proposed architecture and the conventional architectures [1, 9–14] (with dual memory banks), and [15–19]. Table 2 shows comparisons with other commonly used architectures [1, 12–14, 20–24]. For 8×8 DCT, the algorithm proposed by Feig requires 54 multiplications and 462 additions [27]; the proposed method requires 25 multiplications and 100 additions. Thus, the performance of this work is superior to that of the Feig algorithm. In addition, the proposed fast DCT/IDCT architecture is highly regular, scalable, and flexible. The DCT/IDCT processor designed by using the portable and reusable Verilog is a reusable IP, which can be implemented in various processes; combined with efficient use of hardware resources for tradeoffs of performance, area and power consumption; and therefore is much suited to the JPEG and MPEG-1/2 applications.

Acknowledgments

The National Science Council of Taiwan, Taipei, Taiwan, under Grant NSC98-2221-E-216-037 and the Chung Hua University, Hsinchu, Taiwan, under Grant no. CHU-NSC98-2221-E-216-037 supported this work.

References

[1] T.-Y. Sung, "Memory-efficient and high-performance 2-D DCT and IDCT processors based on CORDIC rotation," *WSEAS Transactions on Electronics*, vol. 3, no. 12, pp. 565–574, 2006.

[2] M. Li and W. Zhao, "Representation of a stochastic traffic bound," *IEEE Transactions on Parallel and Distributed Systems*, preprint.

[3] Ming Li, "Fractal time series—a tutorial review," *Mathematical Problems in Engineering*, vol. 2010, Article ID 157264, 26 pages, 2010.

[4] M. Li and S. C. Lim, "Modeling network traffic using generalized Cauchy process," *Physica A*, vol. 387, no. 11, pp. 2584–2594, 2008.

[5] C. Cattani, "Harmonic wavelet approximation of random, fractal and high frequency signals," *Telecommunication Systems*, vol. 43, no. 3-4, pp. 207–217, 2010.

[6] E. G. Bakhoum and C. Toma, "Mathematical transform of traveling-wave equations and phase aspects of quantum interaction," *Mathematical Problems in Engineering*, vol. 2010, Article ID 695208, 15 pages, 2010.

[7] M. Li, "Generation of teletraffic of generalized Cauchy type," *Physica Scripta*, vol. 81, no. 2, Article ID 025007, 2010.

[8] M. Li and J.-Y. Li, "On the predictability of long-range dependent series," *Mathematical Problems in Engineering*, vol. 2010, Article ID 397454, 9 pages, 2010.

[9] T. Y. Sung, "VLSI parallel and distributed computation algorithms for DCT processors," in *Proceedings of the IEEE International Phoenix Conference on Computer and Communications*, pp. 121–125, Scottsdale, Ariz, USA, 1990.

[10] T. Y. Sung, "VLSI parallel and distributed processing algorithms for multidimensional discrete cosine transforms," in *Proceedings of the the Two-Track International Conference on Databases, Parallel Architectures, and Their Applications*, pp. 36–39, Miami Beach, Fla, USA, March 1990.

[11] T. Y. Sung, "Novel parallel VLSI Architectures for discrete cosine transforms," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. 998–1001, Albuquerque, New Mexico, USA, April 1990.

[12] T. Y. Sung and Y. H. Sung, "A novel implementation of cost-effective parallel-pipelined 8 × 8 DCT processor," in *Proceedings of the 4th IEEE Asia-Pacific Conference on Advanced System Integrated Circuits (AP-ASIC '04)*, pp. 200–203, Fukuoka, Japan, August 2004.

[13] T. Y. Sung, Y. S. Shieh, and H. C. Hsin, "Memory efficiency and high-speed architectures for forward and inverse DCT with multiplierless operation," in *Proceedings of the Advances in Image and Video technology*, vol. 4319 of *Lecture Notes in Computer Science*, pp. 802–811, Springer, Berlin, Germany, December 2006.

[14] T. Y. Sung, Y. S. Shieh, and H. C. Hsin, "High-efficiency and low-power architectures for 2-D DCT and IDCT based on CORDIC rotation," in *Proceedings of the 7th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT '06)*, pp. 191–196, December 2006.

[15] Y. H. Hu and Z. Wu, "An efficient CORDIC array structure for the implementation of discrete cosine transform," *IEEE Transactions on Signal Processing*, vol. 43, no. 1, pp. 331–336, 1995.

[16] H. Jeong, J. Kim, and W.-K. Cho, "Low-power multiplierless DCT architecture using image data correlation," *IEEE Transactions on Consumer Electronics*, vol. 50, no. 1, pp. 262–267, 2004.

[17] D. Gong, Y. He, and Z. Gao, "New cost-effective VLSI implementation of a 2-discrete cosine transform and its inverse," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 4, pp. 405–415, 2004.

[18] V. Dimitrov, K. Wahid, and G. Jullien, "Multiplication-free 8 × 8 2D DCT architecture using algebraic integer encoding," *Electronics Letters*, vol. 40, no. 20, pp. 1310–1311, 2004.

[19] M. Alam, W. Badawy, and G. Jullien, "A new time distributed DCT architecture for MPEG-4 hardware reference model," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 5, pp. 726–730, 2005.

[20] Y. P. Lee, T. H. Chen, L. G. Chen, and C. W. Ku, "A cost-effective architecture for 8×8 two-dimensional DCT/IDCT using direct method," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 459–467, 1997.

[21] Y.-T. Chang and C.-L. Wang, "New systolic array implementation of the 2-D discrete cosine transform and its inverse," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 2, pp. 150–157, 1995.

[22] S.-F. Hsiao and W.-R. Shiue, "A new hardware-efficient algorithm and architecture for computation of 2-D DCTs on a linear array," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 11, pp. 1149–1159, 2001.

[23] S.-F. Hsiao and J.-M. Tseng, "New matrix formulation for two-dimensional DCT/IDCT computation and its distributed-memory VLSI implementation," *IEE Proceedings. Vision, Image and Signal Processing*, vol. 149, no. 2, pp. 97–107, 2002.

[24] H. S. Hou, "A fast recursive algorithm for computing the discrete cosine transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 10, no. 35, pp. 1455–1461, 1987.

[25] S.-F. Hsiao, W.-R. Shiue, and J.-M. Tseng, "Design and implementation of a novel linear-array DCT/IDCT processor with complexity of order $I\log_2 N$," *IEE Proceedings. Vision, Image and Signal Processing*, vol. 147, no. 5, pp. 400–408, 2000.

[26] Z. Cvetkovic and M. V. Popovic, "New fast recursive algorithms for the computation of discrete cosine and sine transforms," *IEEE Transactions on Signal Processing*, vol. 40, no. 8, pp. 2083–2086, 1992.

[27] E. Feig and S. Winograd, "Fast algorithms for the discrete cosine transform," *IEEE Transactions on Signal Processing*, vol. 40, no. 9, pp. 2174–2193, 1992.

[28] N. I. Cho and S. U. Lee, "Fast algorithm and implementation of 2-D discrete cosine transform," *IEEE transactions on circuits and systems*, vol. 38, no. 3, pp. 297–305, 1991.

[29] I. Koren, *Computer Arithmetic Algorithm*, chapter 5, A. K. Peters, Natick, Mass, USA, 2nd edition, 2005.

[30] T.-Y. Sung and H.-C. Hsin, "Design and simulation of reusable IP CORDIC core for special-purpose processors," *IET Computers and Digital Techniques*, vol. 1, no. 5, pp. 581–589, 2007.

[31] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins Studies in the Mathematical Sciences, chapter 6, Johns Hopkins University Press, Baltimore, Md, USA, 3rd edition, 1996.

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

[33] "TSMC 0.18 CMOS Design Libraries and Technical Data, v.5.1," Taiwan Semiconductor Manufacturing Company (TSMC), Hsinchu, Taiwan, and National Chip Implementation Center (CIC), National Science Council, Hsinchu, Taiwan, 2009.

[34] Cadence design systems, http://www.cadence.com/products/pages/default.aspx.