논문 2006-43SD-3-7

DA구조 이용 가산기 수를 감소한 2-D DCT/IDCT 프로세서 설계

(2-D DCT/IDCT Processor Design Reducing Adders in DA Architecture)

정 동 윤*, 서 해 준**, 배 현 덕***, 조 태 원***

(Dong-Yun Jeong, Hae-Jun Seo, Hyeon-Deok Bae, and Tae-Won Cho)

요 약

본 논문은 가산기 기반 DA(Distributed Arithmetic: 분산 산술연산)구조로서 ROM과 같은 일반적인 메모리가 사용되지 않는 8×8의 2차원 DCT(Discrete Cosine Transform)/IDCT(Inverse DCT) 프로세서를 제안 설계하였다. 제안된 논문은 DCT와 IDCT의 계수 행렬에서 하드웨어를 줄이기 위해 계수 행렬의 홀수 부분을 공유하였고, 2차원 DCT/IDCT 프로세서의 계수 연산을 위해 단지 29개의 가산기만을 사용하였다. 이는 8×8 1차원 DCT NEDA(NEw DA)구조[5][6][7]에서의 가산기 수 보다 48.6%를 감소 시켰다. 또한, 기존의 전치메모리와는 다른 새로운 전치네트워크 구조를 제안하였다. 제안된 전치네트워크 구조에서는 전치메모리 블록 대신 하드웨어를 줄이기 위해 레지스터 형태의 새로운 레지스터 블록 전치네트워크 형태를 제안하였다. 제안된 전치네트워크 블록은 64개의 레지스터를 사용하며, 이는 일반적인 메모리를 사용하는 기존의 전치메모리 구조에 사용된 트랜지스터 수 보다 18%가 감소하였다. 또한 처리율 향상을 위해 새롭게 적용되고 있는 방식으로, 입력 데이터에 대해 매 클럭 주기마다 8개의 화소데이터를 받아서 8개의 화소데이터를 처리하도록 하여 출력하는 비트 병렬화 구조로 설계하였다.

Abstract

This paper presents 8×8 two dimensional DCT/IDCT processor of adder-based distributed arithmetic architecture without applying ROM units in conventional memories. To reduce hardware cost in the coefficient matrix of DCT and IDCT, an odd part of the coefficient matrix was shared. The proposed architecture uses only 29 adders to compute coefficient operation in the 2-D DCT/IDCT processor, while 1-D DCT processor consists of 18 adders to compute coefficient operation. This architecture reduced 48.6% more than the number of adders in 8×8 1-D DCT NEDA architecture[5][6][7]. Also, this paper proposed a form of new transpose network which is different from the conventional transpose memory block. The proposed transpose network block uses 64 registers with reduction of 18% more than the number of transistors in conventional memory architecture. Also, to improve throughput, eight input data receive eight pixels in every clock cycle and accordingly eight pixels are produced at the outputs.

Keywords : DCT/IDCT, NEDA, Bit-Parallel, Transpose Network

I. 서 론

현재 멀티미디어 시스템에서 최적의 영상압축 기술이며, 디지털 방송에 필수적인 영상신호처리 분야의 핵심기술로 데이터의 압축 및 복원의 효율을 높여주는 알고리듬인 DCT(Discrete Cosine Transform: 이산 여현변환) 및 IDCT(Inverse DCT)는 가장 효과적인 방식으

* 정회원, 매그나칩 반도체 DDI 3팀
(Magnachip Semiconductor, DDI 3Team.)
** 학생회원, *** 평생회원, 충북대학교 전기전자컴퓨터 공학부
(School of Electrical & Electronics Engineering, Chungbuk University)
※ 본 연구는 반도체설계교육센터(IDEC)와 한국소프트웨어 진흥원(IT-SoC)의 연구지원을 받았음.
접수일자: 2005년2월24일, 수정완료일: 2006년3월7일

로 꼽히고 있다.[1] DCT는 KLT(Karhunen-Loeve Transform)에 가장 근접한 성능을 가지고 있으며 정보가 한곳으로 집중되는 효과가 매우 크고, 영상정보를 주파수 영역의 계수(coefficient) 데이터로 변환시켜 낮은 주파수대로 에너지를 집중시키는 등의 압축이 쉬운 형태로 변환, 응용시스템의 압축 효율을 크게 높여줌으로서 HDTV등 응용제품의 소형화에도 유리하다.[2] 따라서, 고속이면서도 적은 하드웨어를 갖는 DCT/IDCT 프로세서가 요구되어 지고, 이를 위해선 2차원 DCT/IDCT 시스템이 적용되어야 한다. 2차원 DCT/IDCT의 효과적인 구현을 위한 기존의 많은 알고리듬과 시스템 구조들이 제안 되어 졌는데, 이들은 수행방법에 따라 크게 행렬 순차수행(row-column decomposition)과 2차원 직접수행(direct 2-D)방법으로 나눌 수 있다. 이 중 2차원 직접수행 방법은 하드웨어가 복잡하기 때문에 하드웨어가 간단한 행렬 순차수행 방법에 관하여 더 많은 연구가 진행되어 왔다. 행렬 순차수행 방법은 입력 데이터의 행렬에 대하여 행(혹은 열)에 대한 1-D DCT/IDCT를 먼저 수행하고 그 결과의 행렬을 전치(trans-pose)한 다음, 열(혹은 행)에 대한 1-D DCT/IDCT를 수행하는 것이다.[3][4] 그림1에서와 같이 최근 수행방법이 행렬순차 방법이면서 행렬계수의 변형을 통한 다양한 구조의 방식이 새롭게 연구되고 있는데, 이 중 DA(Distributed Arithmetic: 분산산술연산)방식[5][6][7]은 ROM기반(ROM based)[8]과 가산기 기반(adder based)[9]으로 나눠지며, 또한 가산기 기반의 방법은 구현 방식에 따라 2의 보수(2's complement)와 CSD(Canonical Signed Digit)방식으로 구현될 수 있다. 이는 다른 방식에 비해 하드웨어의 구현에 있어서 면적에 상당한 절감의 효과가 있으며 구현에 있어서 수월하기 때문에 많이 연구되고 있다.

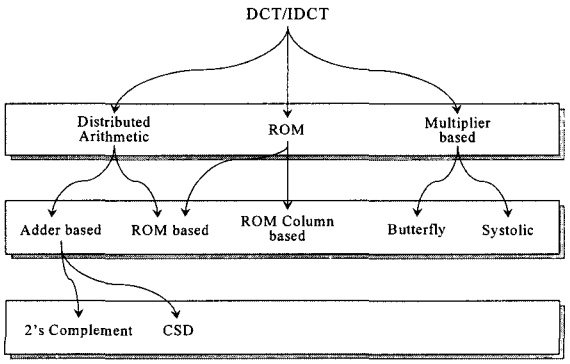


그림 1. DCT/IDCT 구현방법에 따른 분류
Fig. 1. Distributary by DCT/IDCT implementation method.

따라서 본 논문에서는 행렬순차 수행방법이면서 2's complement를 사용한 가산기기반의 DA방식의 DCT/IDCT 프로세서를 제안 설계하였다. 또한 입력을 bit-parallel하게 처리하도록 하고 계수를 분산하여 주어서 가산기의 수를 감소시켰으며, 처리율을 증가하도록 하였다. 본 논문에서 설계한 2차원 DCT/IDCT 프로세서를 구현하기 위해 DCT와 IDCT 각각의 홀수 부(odd part)의 행렬 계수를 공유함으로서 하드웨어를 감소시켰다. 뿐만 아니라, 일반적인 2차원 DCT/IDCT 프로세서의 설계를 위해 사용되는 전치 메모리블록(transpose memory block)에서 새로운 구조의 전치 네트워크 형태를 제안하여 설계함으로서 효율성을 극대화하였다.

본 논문의 구성은 제Ⅱ장에서는 DCT/IDCT의 알고리듬에 대해 설명한다. 제Ⅲ장에서는 본 논문에서 제안한 알고리듬과 전체 시스템인 2차원 DCT/IDCT 프로세서의 구현을 위해 각각의 세부 기능 블록들에 대해서 설명을 하고, 제Ⅳ장에서는 제안한 2차원 DCT/IDCT 프로세서와 다른 연구논문의 결과 비교 및 특징을 기술하고 마지막 제Ⅴ장에서는 결론 및 고찰과 향후 연구방향에 대해 기술한다.

Ⅱ. DCT/IDCT 알고리즘

본 장에서는 DCT/IDCT 알고리듬 개요와 특징을 기술한다. 또한 2차원 DCT/IDCT의 알고리듬을 1차원 DCT/ IDCT의 알고리듬으로 정규화 시키는 수식 과정을 기술하고, 각각의 1차원 DCT/IDCT 정규화된 알고리듬 행렬식을 4×4행렬의 짝수 부(Even Part)/홀수 부(Odd Part)의 행렬식으로 분해하는 과정을 기술한다.

1. 2차원 DCT/IDCT 알고리듬 개요

2차원 DCT/IDCT의 기본적인 구조는 아래의 그림 2와 같이 두 개의 1차원 DCT/IDCT 블록과 전치네트워크 및 제어부 블록으로 구성된다.

그림 2에서 입력 화소의 값을 1차원 DCT/IDCT를 수행하고, 그 결과를 전치 네트워크에 저장한다. 그리고 저장된 데이터를 행(열)과 열(행)을 바꾸어서 다시 1차원 DCT/IDCT 연산을 수행하면 2차원 DCT/IDCT의 결과 값을 얻을 수 있게 된다.[2]

정규화 된 N×N 블록의 2차원 DCT 식은 다음의 식(1)과 같고, $X_{(i,j)}$는 2차원 입력 화소(pixel) 데이터이고, $Y_{(u,v)}$는 2차원으로 변환된 데이터이다.

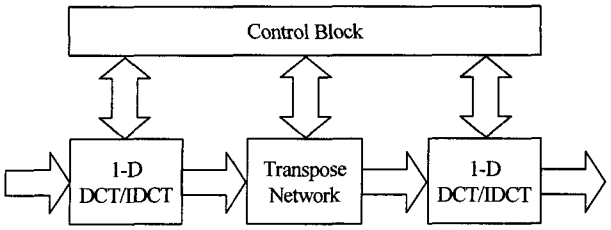


그림 2. 일반적인 2차원 DCT/IDCT
Fig. 2. Conventional 2-D DCT/IDCT.

$$Y_{(u,v)} = \frac{2}{N} A(u)A(v) \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} X_{(i,j)} \cdot \cos\frac{(2i+1)u\pi}{2N} \cos\frac{(2j+1)v\pi}{2N}$$
$$A(0) = \frac{1}{\sqrt{2}}, \; A(u) = A(v) = 1 \; (u \neq 0, v \neq 0). \tag{1}$$

정규화 된 N×N 블록의 2차원 IDCT 식은 다음 식(2)와 같다.

$$X_{(i,j)} = \frac{2}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} A(u)A(v) Y_{(u,v)} \cdot \cos\frac{(2i+1)u\pi}{2N} \cos\frac{(2j+1)v\pi}{2N}$$
$$A(0) = \frac{1}{\sqrt{2}}, \; A(u) = A(v) = 1 \quad (u \neq 0, v \neq 0). \tag{2}$$

식(1)과 식(2)를 행-열 분해(row-column decomposition)에 의한 행렬 형태로 표현하면 다음 식(3),(4)와 같다.[4] 2차원 DCT인 경우 첫 번째 1차원 DCT/IDCT 블록에서는 식(3) 중에서 $y = CX^T$ 를 수행하고, 두 번째 1차원 DCT/IDCT 블록에서는 y 를 전치하여 $Y = Cy^T$ 를 수행한다. 2차원 IDCT인 경우 첫 번째 1차원 DCT/IDCT 블록에서는 $y = C^T Y^T$ 를 수행하고, 두 번째 1차원 DCT/IDCT 블록에서는 y 를 전치하여 $X = C^T y^T$ 를 수행하게 된다. 즉, DCT의 기저벡터는 C 가 되며, IDCT의 기저벡터는 C^T 가 되며, 따라서 DCT 와 IDCT는 기저벡터 C 가 전치되었을 뿐 동일한 구조이다.

$$Y = CXC^T = C[CX^T]^T \tag{3}$$

$$X = C^T YC = C^T[C^T Y^T]^T \tag{4}$$

행렬식(3),(4)에서 C 는 N×N 행렬로써 C 의 원소 $C_{(k,n)}$은 아래의 식(5)로 나타내고, 각 N×N 행렬의 곱은 N 행렬-벡터의 곱으로 나누어진다.

$$C_{(k,n)} = \sqrt{\frac{2}{N}} A(k) \cos\frac{(2n+1)k\pi}{2N}, \quad \text{for } n, k = 0, 1, \ldots, N-1$$
$$A(k) = \frac{1}{\sqrt{2}}, \quad k = 0$$
$$A(k) = 1, \quad otherwise. \tag{5}$$

또한 식(5)의 기저벡터 C 는 식(6)과 같이 나타낼 수 있다.

$$C = \frac{1}{2}\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ C_1 & C_3 & C_5 & C_7 & C_9 & C_{11} & C_{13} & C_{15} \\ C_2 & C_6 & C_{10} & C_{14} & C_{18} & C_{22} & C_{26} & C_{30} \\ C_3 & C_9 & C_{15} & C_{21} & C_{27} & C_{33} & C_{39} & C_{45} \\ C_4 & C_{12} & C_{20} & C_{28} & C_{36} & C_{44} & C_{52} & C_{60} \\ C_5 & C_{15} & C_{25} & C_{35} & C_{45} & C_{55} & C_{65} & C_{75} \\ C_6 & C_{18} & C_{30} & C_{42} & C_{54} & C_{66} & C_{78} & C_{90} \\ C_7 & C_{21} & C_{35} & C_{49} & C_{63} & C_{77} & C_{91} & C_{105} \end{bmatrix} \tag{6}$$

식(5),(6)에서 $\cos\frac{4\pi}{16} = \frac{1}{\sqrt{2}}$ 이므로 식(6)에서 $\frac{1}{\sqrt{2}}$ 을 C_4로 대치시킬 수 있고, 또한 상수 $\frac{1}{2}$ 을 행렬 속에 포함시켜서 $C_n = \frac{1}{2}\cos\frac{n\pi}{16}$ 로 다시 정의하여 식(7)과 같이 나타낼 수 있다.

$$C = \begin{bmatrix} C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4 \\ C_1 & C_3 & C_5 & C_7 & -C_7 & -C_5 & -C_3 & -C_1 \\ C_2 & C_6 & -C_6 & -C_2 & -C_2 & -C_6 & C_6 & C_2 \\ C_3 & -C_7 & -C_1 & -C_5 & C_5 & C_1 & C_7 & -C_3 \\ C_4 & -C_4 & -C_4 & C_4 & C_4 & -C_4 & -C_4 & C_4 \\ C_5 & -C_1 & C_7 & C_3 & -C_3 & -C_7 & C_1 & -C_5 \\ C_6 & -C_2 & C_2 & -C_6 & -C_6 & C_2 & -C_2 & C_6 \\ C_7 & -C_5 & C_3 & -C_1 & C_1 & -C_3 & C_5 & -C_7 \end{bmatrix} \tag{7}$$

따라서, 식(3),(7)을 이용하여 1차원 DCT $y = CX^T$ 를 식(8)과 같이 행렬식으로 나타낼 수 있다. 식(8)의 출력 $y_j (j = 0, 1, \ldots, 7)$은 입력화소 값 $X_i (i = 0, 1, \ldots, 7)$과 DCT의 계수 값인 $C_n (n = 0, 1, \ldots, 7)$ 로 구성된다.

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4 \\ C_1 & C_3 & C_5 & C_7 & -C_7 & -C_5 & -C_3 & -C_1 \\ C_2 & C_6 & -C_6 & -C_2 & -C_2 & -C_6 & C_6 & C_2 \\ C_3 & -C_7 & -C_1 & -C_5 & C_5 & C_1 & C_7 & -C_3 \\ C_4 & -C_4 & -C_4 & C_4 & C_4 & -C_4 & -C_4 & C_4 \\ C_5 & -C_1 & C_7 & C_3 & -C_3 & -C_7 & C_1 & -C_5 \\ C_6 & -C_2 & C_2 & -C_6 & -C_6 & C_2 & -C_2 & C_6 \\ C_7 & -C_5 & C_3 & -C_1 & C_1 & -C_3 & C_5 & -C_7 \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \\ X_7 \end{bmatrix} \tag{8}$$

또한, 1차원 IDCT는 식(4)에서 $y = C^T Y^T$ 를 식(9)와 같은 행렬식으로 나타낼 수 있으며, 1차원 DCT와 같이 행렬 분해방법을 이용하여 나타낼 수 있다.

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} C_4 & C_1 & C_2 & C_3 & C_4 & C_5 & C_6 & C_7 \\ C_4 & C_3 & C_6 & -C_7 & -C_4 & -C_1 & -C_2 & -C_5 \\ C_4 & C_5 & -C_6 & -C_1 & -C_4 & C_7 & C_2 & C_3 \\ C_4 & C_7 & -C_2 & -C_5 & -C_4 & C_3 & -C_6 & -C_1 \\ C_4 & -C_7 & -C_2 & C_5 & C_4 & -C_3 & -C_6 & C_1 \\ C_4 & -C_5 & -C_6 & C_1 & C_4 & -C_7 & C_2 & -C_3 \\ C_4 & -C_3 & C_6 & C_7 & -C_4 & C_1 & -C_2 & C_5 \\ C_4 & -C_1 & C_2 & -C_3 & C_4 & -C_5 & C_6 & -C_7 \end{bmatrix} \begin{bmatrix} Y_0 \\ Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ Y_5 \\ Y_6 \\ Y_7 \end{bmatrix} \quad (9)$$

1차원 DCT와 IDCT의 행렬 분해방법을 이용하여 나타낸 식(8)과 (9)를 행렬의 교환특성을 이용하여 짝수부와 홀수부로 정규화 하여 나타 낼 수 있다.[10] 이러한 DCT/IDCT 변환의 특징은 첫째, 빠른 알고리듬을 위해서는 계수 행렬들을 분해한다. 둘째, 단일 변환이므로 변환 전후의 에너지의 합은 일정하다. 셋째, 에너지 압축률이 매우 높기 때문에 변환 후에는 몇 개의 계수에 대부분의 값이 압축된다. 따라서 여러 가지 영상 변환 중에서 최적의 변환이라고 알려진 KLT변환의 성능에 가장 근접한 변환이라 할 수 있다.

III. 제안한 2차원 DCT/IDCT 프로세서

이번 장에서는 본 논문에서 제안한 가산기 기반 DA 방식의 2차원 DCT/IDCT 프로세서의 구조와 알고리듬을 기술한다. 제안한 구조는 가산 공유 항을 최소화하였으며 DCT와 IDCT가 선택적으로 동작이 가능하고 계수의 Odd Part를 공유함으로서, 기존의 구조를 개선하였다. 또한 기존의 2차원 DCT/IDCT 프로세서들은 전치메모리에서 면적이 커지고 제어하기 어려운 RAM과 같은 일반적인 메모리를 사용하는 반면에, 본 논문에서는 새로운 전치네트워크 구조를 제안하여 일반적인 메모리 장치를 사용하지 않고 레지스터로 구현하였다.

1. 알고리듬 개요

제안한 2차원 DCT/IDCT 프로세서는 2장에서 설명한 바와 같이 DCT/IDCT를 1차원 DCT/IDCT로 정규화 한 후, 각각의 DCT/IDCT의 계수 행렬을 4×4의 짝수/홀수 행렬인 Even/Odd Part로 나누어 수행한다. 1차원 DCT/IDCT의 Even/Odd Part의 식을 살펴보면 그림 3과 같다.

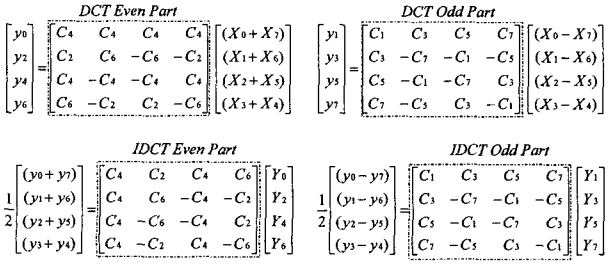


그림 3. DCT/IDCT의 4×4 Even/Odd 행렬
Fig. 3. 4×4 Even/Odd matrix of DCT/IDCT.

그림 3에서 보면 DCT의 Odd Part와 IDCT의 Odd Part의 계수 행렬식이 같음을 알 수 있다. 따라서 DCT/IDCT 프로세서를 구현함에 있어서 DCT와 IDCT의 Odd Part의 계수 행렬을 공유함으로서 하드웨어 면적을 감소시켰다. 공유된 Odd Part를 수행하기 위해 DCT/IDCT의 입력과 출력부분에 MUX(Multiplexer: 다중화기)와 DE MUX(Demultiplexer: 역다중화기)를 사용하여 수행한다. 또한 4×4 계수 행렬을 사용함으로서 DCT의 입력 화소 값은 전처리단(pre-computation block)을 구성하여 AD D(Addition: 가산) / SUB (Subtraction: 감산)하여 수행 하여야 하며, IDCT에서는 반대로 계수 식을 수행한 후 각각의 출력 값을 얻기 위해서는 후처리단(post- computation block)에서 ADD/SUB를 수행해야 한다.

2. 2차원 DCT/IDCT 전체 구조

제안한 2차원 DCT/IDCT의 전체 구조도를 그림4에 나타내었다. 그림에서 각각의 DCT/IDCT의 입력은 Mode_ select 신호에 의해서 선택적으로 수행할 수 있다. 각각의 입력 화소 데이터는 bit-parallel하게 수행되며 8×1의 입력 화소데이터가 한 클럭에 입력되어 클럭 동기에 맞춰 출력도 8×1의 화소 데이터가 출력된다. 또한 1차원 DCT/IDCT의 수행된 데이터를 다시 1차원 DCT/IDCT를 수행하기 위해 전치네트워크를 수행하게 되는데 수행 데이터의 오류를 감소하기 위해 16비트의 데이터 패스를 유지하였다. 제안한 2차원 DCT/IDCT 프로세서는 별도의 내부 클럭이 필요하지 않으며 외부의 클럭으로 동작을 수행한다.

그림 5는 제안된 2차원 DCT/IDCT 프로세서의 세부구조를 나타낸 것이며 크게 4개의 블록인 입/출력 블록,

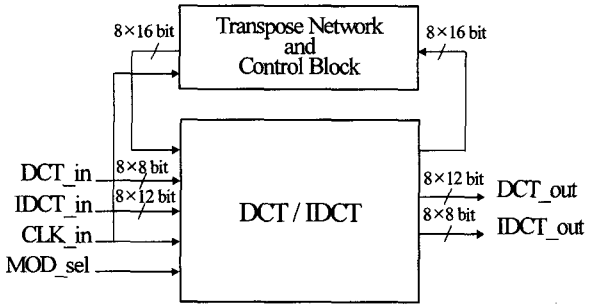


그림 4. 제안한 2차원 DCT/IDCT의 전체 구조
Fig. 4. Total architecture of proposed 2-D DCT/IDCT.

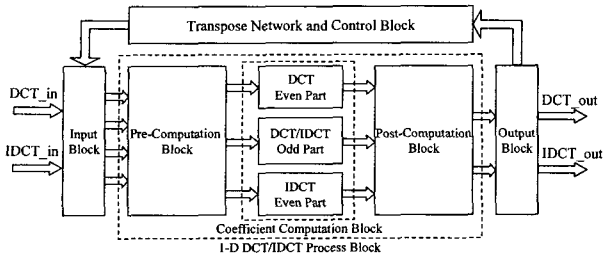


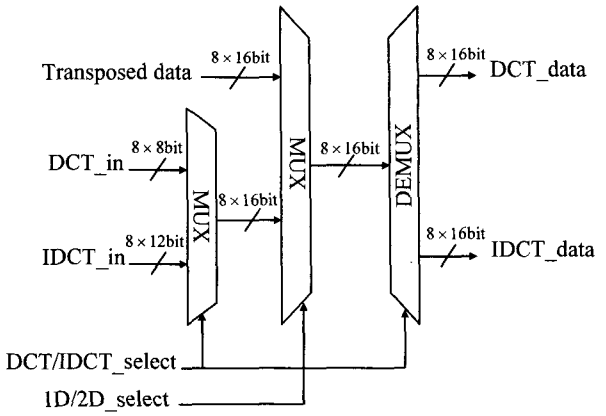
그림 5. 제안된 프로세서의 세부 구조
Fig. 5. Detail architecture of proposed processor.

1차원 DCT/IDCT 처리 블록, 전치네크워크 블록으로 구성 되어있다. 또한 1차원 DCT/IDCT 처리 블록은 전처리단과 계수처리단(coefficient computation block)의 세 가지 블록인 1) DCT의 Even Part, 2) 공유된 DCT/IDCT의 Odd Part를 공유한 부분, 3) IDCT의 Even Part를 가지며, 마지막으로 후처리 단으로 구성된다.

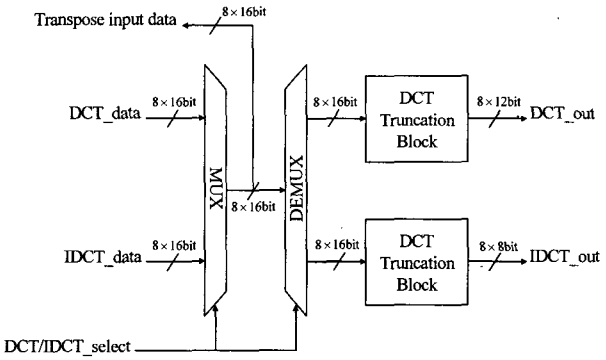
3. 각 블록의 세부구조

가. 입/출력 블록

그림 6의 (a)에서와 같이 각각의 입력 데이터들은 bit- parallel 하게 수행되어진다. 따라서 DCT_in은 8개의 8비트를 갖는 화소 값이고, IDCT_in은 8개의 12비트를 갖는 데이터이며, 전치된 데이터는 8개의 16비트를 갖는 데이터이다. 또한 입력 DCT_in과 IDCT_in은 부호(sign) 비트를 16비트로 확장하였으며 내부데이터의 에러율을 감소하기 위해 16비트의 데이터 패스를 이루도록 하였다. 그림 6(b)의 출력 블록에서 Truncation Block은 8개의 Truncation 레지스터로 이루어져 있으며, ITU-T (ITU_T Standardization Sector)규정에 의해 DCT의 출력 화소 값은 12비트의 데이터로 이루어지며, IDCT의 출력 화소 값은 8비트의 데이터로 이루어진다. 또한, 2차원 DCT/IDCT를 수행함에 있어 전치 네트워크 블록을 수행하기 위해 8개의 16비트 화소데이터가 MUX를 통해 출력된다.



(a) 입력 블록



(b) 출력 블록

그림 6. 입력/출력 블록의 구조
Fig. 6. Architecture of input/output block.

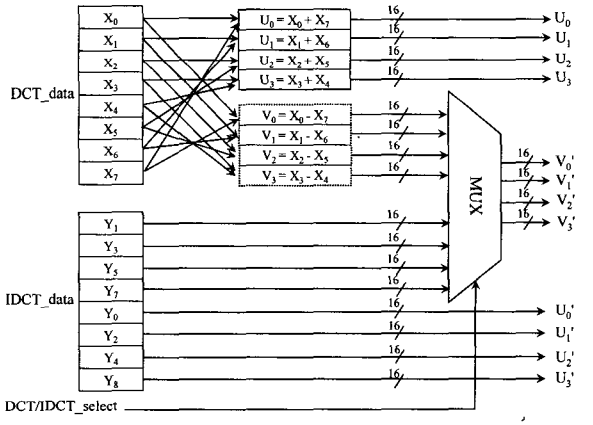


그림 7. 전처리단의 구조
Fig. 7. Architecture of pre-computation block.

나. 전처리단

전처리단은 DCT의 Even Part와 Odd Part의 입력을 수행하는 블록과 DCT/IDCT의 Odd Part를 선택해 주는 MUX로 구성되어 있다. 입력 블록에서 선택된 데이터가 DCT 데이터일 경우 입력 $(X_0 \sim X_7)$를 Even Part와 Odd Part의 입력에 맞추어 그림7에서와 같이 Even Part일 경우는 U_0, U_1, U_2, U_3로 치환해 주어야 하고, Odd Part일 경우는 V_0, V_1, V_2, V_3로 치환을 수행해 주어야 한다.[10] 또한 입력 블록에서 선택된 데이터가 IDCT 데이터일 경우는 데이터의 변환과정이 필요하지 않는다. 그러나 DCT의 전처리단의 과정을 수행한 Odd Part의 데이터 V_0, V_1, V_2, V_3와 IDCT의 Odd Part의 입력 데이터 Y_1, Y_3, Y_5, Y_7은 계수처리단에서의 공유된 DCT/ IDCT Odd Part의 입력과정을 수행하기 위해서 MUX의 과정이 필요하고, IDCT의 Even

Part의 입력데이터 Y_0, Y_2, Y_4, Y_6은 전처리단에서 별도의 과정을 수행하지 않는다.

다. 계수처리단

1) DCT Even Part

그림 8은 계수 처리단에서 DCT의 Even Part의 계수식을 나타낸 것이다. 각각의 계수값 $C_4, -C_4, C_2, -C_2, C_6, -C_6$는 $2^0 \sim 2^{-12}$ 까지의 이진(binary)값 13비트를 가지며 2^0의 값은 부호(sign) 값을 나타낸 것이다. 그림 8의 (1)에서 살펴보면, 각각의 행의 값은 계수값 ($C_4, C_4\ C_4, C_4$)을 나타내며 각각의 열의 값은 계수 값의 가중치에 따른 값을 나타낸 것이다. 이와 같은 방법은 DA에서 입력은 bit-parallel하게 처리하는 반면에, 계수 값을 분산하여 연산하는 방법이다.[5][6][7] 각각의 행렬 계수값은 $C_4 = 0.010110101000$, $C_2 = 0.011101100100$, $C_6 = 0.001100001111$을 가중치에 따라 행으로 나타낸 것이다. 같은 방법으로 그림의 (2),(3),(4)를 나타내었으며, 음수의 계수값 ($-C_4, -C_2, -C_6$)은 2의 보수 값으로 나타낼 때 C_4, C_2, C_6의 이진 값을 반전(inversion)한 값과 같아진다. 또한 가중치가 적용된 계수 값들 중에 가중치가 낮은 값들의 변화는 데이터의 정확도에 큰 영향을 미치지 못하기 때문에 이와 같은 방법을 사용하여 계수처리단의 DCT의 Even Part에 적용하였고, 또한 DCT/IDCT의 Odd Part와 IDCT의 Even Part에도 적용하였다. 전처리단을 거쳐 나온 16비트 입력데이터 U_0, U_1, U_2, U_3 는 DCT의 Even Part 입력이 되며, 각각의 입력을 사용하여 그림 8에서 표의 Addition Type과 같이 입력에 대해 공유된 공유 가산항

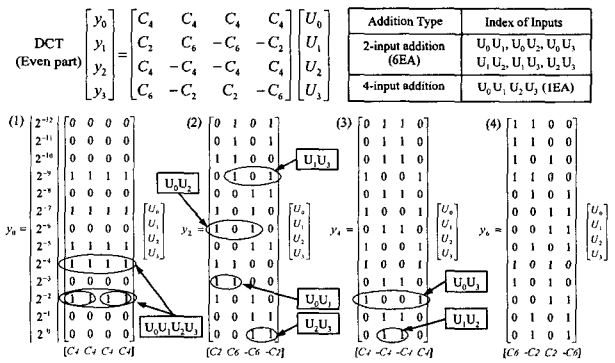


그림 8. DCT의 4×4 Even Part와 공유 가산항
Fig. 8. 4×4 Coefficient even part of DCT and common sharing addition.

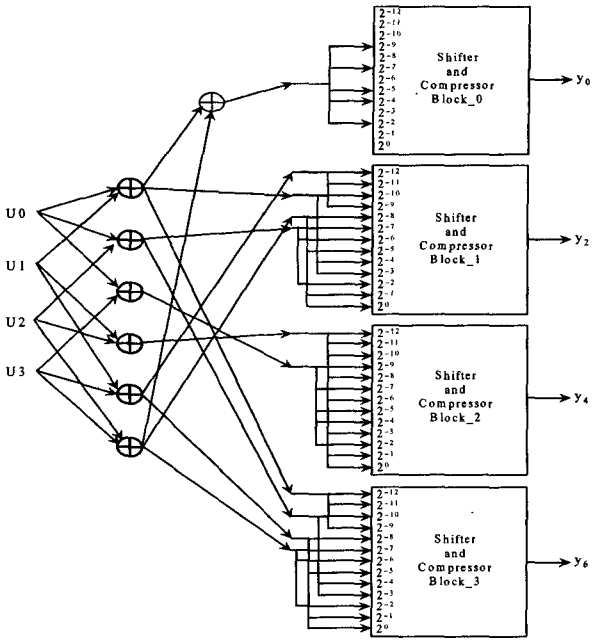


그림 9. DCT Even Part의 구조
Fig. 9. Architecture of DCT even part.

(common sharing addition)을 나타낼 수 있다. 공유 가산항은 2개의 입력 값 (U_0U_1, U_0U_2, U_0U_3, U_1U_2, U_1U_3, U_2U_3)을 가지는 가산기(2-input addition)가 6개로 구성할 수 있다. 또한 4개의 입력 값을 가진 가산기(4-input addition)는 2개의 입력 값을 가진 가산기의 출력 값을 사용하여 구성할 수 있다. 따라서 계수처리를 위한 DCT의 Even Part는 최종적으로 7개의 가산기를 사용하여 수행할 수 있다.

그림 9는 DCT의 Even Part의 세부 블록도를 나타낸 것이다. 그림에서 7개의 가산기와 각각에 해당하는 값의 가중치에 따른 값들을 뒷단에서 쉬프트와 3-level 4:2 압축 단을 사용하여 최종 DCT의 Even Part의 출력 (y_0, y_2, y_4, y_6)을 얻을 수 있다. 전처리단을 수행하여 선택되어 나온 Odd Part의 입출력 데이터는 DCT/IDCT의 Odd Part의 4×4 계수식을 공유하기 때문에 입출력을 식(10)과 같이 바꿀 수 있다.

$$\begin{bmatrix} O_0 \\ O_1 \\ O_2 \\ O_3 \end{bmatrix} = \begin{bmatrix} C_1 & C_3 & C_5 & C_7 \\ C_3 & -C_7 & -C_1 & -C_5 \\ C_5 & -C_1 & C_7 & C_3 \\ C_7 & -C_5 & C_3 & -C_1 \end{bmatrix} \begin{bmatrix} V_0' \\ V_1' \\ V_2' \\ V_3' \end{bmatrix} \quad (10)$$

2) DCT/IDCT Odd Part

DCT/IDCT의 공유된 Odd Part의 식(10)은 다음의 표 1과 같이 정리 할 수 있다. 표 1은 계수처리단에서

표 1. DCT/IDCT Odd Part의 공유된 가산기
Table 1. Common adders of DCT/IDCT Odd Part.

Addition Type	Index of Inputs
2-input addition (6EA)	$V_0'V_1'$, $V_0'V_2'$, $V_0'V_3'$, $V_1'V_2'$, $V_1'V_3'$, $V_2'V_3'$
3-input addition (4EA)	$V_0'V_1'V_2'$, $V_0'V_1'V_3'$, $V_0'V_2'V_3'$, $V_1'V_2'V_3'$
4-input addition	$V_0'V_1'V_2'V_3'$ (1EA)

공유된 DCT/IDCT의 4×4 Odd Part의 계수식과 공유 가산항을 나타낸 것이다. DCT의 Even Part와 같은 방식으로 수행을 하며, 비트 별로 나타낸 계수값의 열의 non-zero 비트가 DCT의 Even Part와는 달리 홀수개가 발생하기 때문에 3입력 가산기가 필요하며, 열의 non-zero 비트가 1개일 경우는 뒷단의 압축단의 입력으로 라우팅하여 주면 된다. Odd Part를 수행하기 위해서는 표 1에서 보는 것과 같이 11개의 가산기로서 수행할 수 있다. 따라서 만약 8×8 DCT를 구현하기 위해서는 전체 18개의 가산기로서 구현 가능하다. 표 1의 공유된 DCT/IDCT의 Odd Part는 DCT의 Even Part와 동일한 과정으로 수행함으로서 Odd Part의 출력값 (O_0, O_1, O_2, O_3)을 얻을 수 있으며, 식(10)으로 치환해 준 DCT와 IDCT의 각각의 출력의 값에 대한 수행은 후처리단에서 하였다.

3) IDCT Even Part

표 2는 계수처리단의 IDCT의 4×4 Even Part와 공유 가산항을 나타낸 것이다. 수행방법은 DCT의 Even Part와 공유된 DCT/IDCT의 Odd Part와 같으며 11개의 가산기로서 수행할 수 있다.

표 2. IDCT Even Part의 공유된 가산기
Table 2. Common adders of IDCT even part.

Addition Type	Index of inputs
2-input Addition (6EA)	$U_0'U_1'$, $U_0'U_2'$, $U_0'U_3'$, $U_1'U_2'$, $U_1'U_3'$, $U_2'U_3'$
3-input Addition (4EA)	$U_0'U_1'U_2'$, $U_0'U_1'U_3'$, $U_0'U_2'U_3'$, $U_1'U_2'U_3'$
4-input Addition	$U_0'U_1'U_2'U_3'$ (1EA)

라. 후처리단

그림 10은 후처리단을 나타낸 것이며 DEMUX와 4

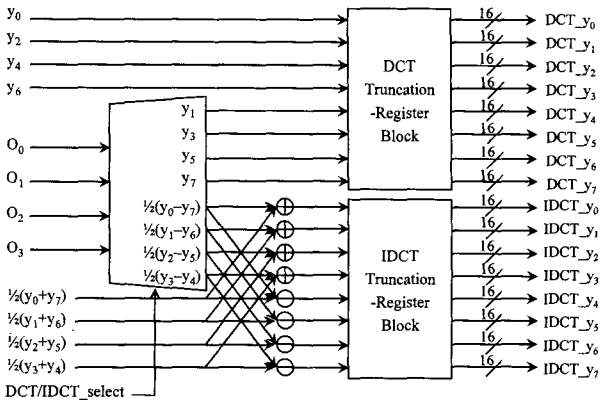


그림 10. 후처리단의 구조
Fig. 10. Architecture of post-computation block.

개의 가산기와 4개의 감산기 그리고 DCT/IDCT Truncation 레지스터 블록으로 이루어 졌다. 계수 처리단을 수행한 DCT의 Even Part의 결과 값 (y_0, y_2, y_4, y_6)과 공유된 DCT/IDCT Odd Part의 결과값 (O_0, O_1, O_2, O_3) IDCT의 Even Part의 결과 값 $(\frac{1}{2}(y_0+y_7), \frac{1}{2}(y_1+y_6), \frac{1}{2}(y_2+y_5), \frac{1}{2}(y_3+y_4))$은 후처리단의 입력이 된다. 또한 그림 10에서 Odd Part 데이터는 DEMUX의 DCT/IDCT_select 신호에 따라 '0' 일 때는 DCT의 Odd Part의 데이터로 '1' 일 때는 IDCT의 Odd Part의 데이터가 된다. DCT의 데이터 $(y_0 \sim y_7)$은 다음 블록인 DCT Truncation 레지스터 블록을 수행하게 되며 상위 16비트의 값을 가진 유효 데이터를 취하고 가중치가 적은 나머지 하위데이터는 버린다. 또한 IDCT의 데이터는 IDCT의 Odd Part의 결과 값 $(\frac{1}{2}(y_0-y_7), \frac{1}{2}(y_1-y_6), \frac{1}{2}(y_2-y_5), \frac{1}{2}(y_3-y_4))$과 IDCT의 Even Part의 결과 값을 각각 ADD/SUB 한 후 Truncation하여 IDCT의 결과 값 $(y_0 \sim y_7)$을 얻을 수 있다.

마. 전치 네트워크 및 제어 블록

1) 전치 네트워크부

본 논문에서는 기존 방식과 다른 새로운 형태인 64개의 레지스터를 사용하여 전치네트워크를 제안하여 면적과 제어부분에서도 더 효과적이고 간단하도록 구현하였다. 그림 11은 전치네트워크 블록을 나타낸 것으로, 64개의 레지스터로 이루어진 레지스터 블록과 64입력 8출력의 MUX로 이루어져 있다. 그림에서 보면 매 클럭마

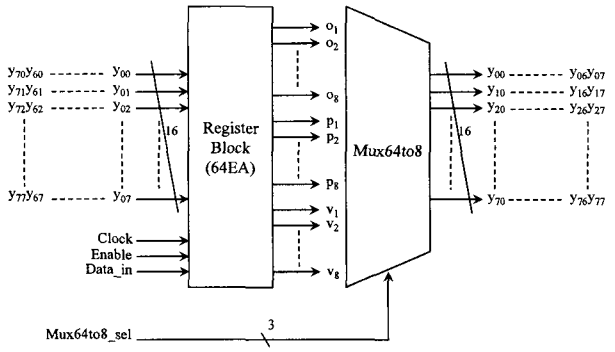


그림 11. 전치네트워크의 구조
Fig. 11. Architecture of transpose network block.

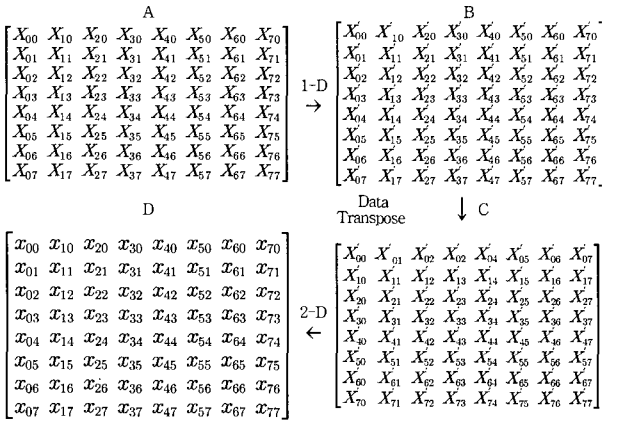


그림 12. 전치네트워크의 데이터 수행과정
Fig. 12. Data operation processing of transpose network.

다 1차원의 결과 값 8×1의 화소 값을 받아들여 레지스터 블록에서 쉬프트 한 다음 8클럭이 끝나면 8×8의 64개의 화소를 레지스터에 저장한 다음 Data_in과 Enable 신호에 의해서 레지스터의 값을 읽어준다. 이후 64입력 8출력의 MUX에서 Mux64to8_sel의 신호에 의해서 9클럭 때에 전치된 8×1의 화소 값을 출력한다. 따라서 전치를 수행하기 위해서는 8클럭이 필요하게 된다. 1차원 DCT/IDCT를 수행한 결과 값 64개의 화소 값을 전치하여 2차원 DCT/IDCT의 입력 데이터가 된다.

그림 12는 전치 네트워크의 데이터 수행과정을 나타낸 것이다. 그림에서 보는 것과 같이 1차원 DCT/IDCT 수행을 거치고 난 후 B과정에서 C과정으로 데이터 들이 전치되어서 다시 2차원 DCT/IDCT를 수행한다.

2) 제어부

그림 13의 제어부는 전체 2차원 DCT/IDCT 프로세서를 제어하는 신호를 발생하는 제어 블록이며, 기존의 일반적인 제어부에 비해 매우 간단히 제어 할 수 있는 장점이 있다. 제어부는 전치네트워크에 필요한 제어신

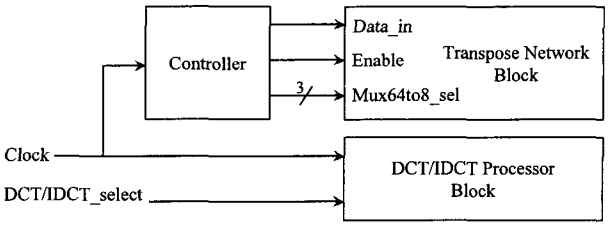


그림 13. 제어부의 구조
Fig. 13. Architecture of controller block.

호를 발생해 준다. 전치네트워크 블록에서는 1차원 DCT/ IDCT를 수행한 8×1의 화소값 8개를 매 클럭마다 읽어 들여 8클럭동안 64개의 화소값을 읽어들이는 Data_in 신호가 필요하며, 읽어 들인 데이터를 전치네트워크 블록을 수행한 데이터 64개의 화소 값을 Enable 하는 신호와 Enable된 화소 값을 전치 네크워크의 64입력 8출력의 MUX를 통해 8×1의 8개의 화소 값을 출력하기 위한 MUX의 선택 신호인 MUX64to8_sel의 3비트 신호가 필요하며, 이 3비트 신호는 Counter로 구성되어 있다.

Ⅳ. 결과 및 분석

본 논문에서의 설계검증은 하드웨어 기술언어(VHDL)로 기술하여 모의실험을 하였으며, Compass 툴을 사용하여 0.6㎛ CMOS 공정 라이브러리로 합성하였다.

1. 전체 구조의 특성과 합성결과

표 3은 본 논문에서 설계한 구조의 결과로 기존의 DA구조의 방식으로 곱셈기나 롬(ROM)을 사용하여 설계된 2차원 DCT/IDCT의 구조방식에 비해 하드웨어 면적면에서 효율성이 크며, 또한 2차원 DCT/IDCT의 구현시 DCT와 IDCT의 Odd Part를 공유하여 설계함으로서 하드웨어의 절감 효과를 더욱 향상 시켰다.

표 3. 전체 구조의 특성
Table 3. Characteristics of total architecture.

	DCT	IDCT
Input bits	8	12
Output bits	12	8
Technology	0.6㎛ CMOS	
Gate counts	94,500	
Transistor counts	378,000	
Throughput	128Mpixels/s	

2. 성능 비교

가. 기존의 NEDA구조와 비교

제안된 구조의 전체 시스템인 2차원 DCT/IDCT를 수행하기 위한 핵심 프로세서인 1차원 DCT/IDCT를 수행하기 위해서는 DCT의 Even Part에서 7개의 가산기, 공유된 DCT/IDCT의 Odd Part에서 11개의 가산기와 IDCT의 Even Part에서 11개의 가산기로서 전체 29개의 가산기를 사용하여 1차원 DCT/IDCT 프로세서를 수행할 수 있었다. 표 4는 제안된 구조와 기존의 NEDA(NEw Distributed Arithmetic)구조[5][6][7]와의 1차원 DCT의 가산기의 수를 비교한 것이다. 제안된 구조는 입력데이터를 bit-parallel하게 처리하는 NEDA구조의 방식과 동일한 방식으로 데이터를 처리하였으며, 기발표된 NEDA구조[6][7]와 비교 대상을 동일시하여 1차원 DCT 만을 비교하여 보았을 때 계수 처리 단에서의 가산기의 수를 35개에서 18개로 줄임으로서 48.6%의 절감 효과를 얻을 수 있었다.

표 4. 제안된 구조와 NEDA 구조의 비교
Table 4. Comparison of proposed architecture and NEDA architecture.

	NEDA		Proposed Structure				
	1D-DCT [5][6]	1D-DCT /IDCT [7]	1D-DCT		1D-DCT/IDCT		
Coefficient Computation (addition counts)	35	97	DCT Even Part	Common DCT/IDCT Odd Part	DCT Even Part	Common DCT/IDCT Odd Part	IDCT Even Part
			7	11	7	11	11
			18		29		
Saving[%]	64		48.6				

표 5. 제안한 구조와 기존 구조의 성능 비교
Table 5. Comparison of proposed and conventional architecture.

	Chen S. Kim [8]	J.R. Choi [11]	Y.P. Lee [12]	Proposed
Function	2-D IDCT	2-D IDCT	2-D DCT/IDCT	2-D DCT/IDCT
Realization approach	ROM-based	Radix-two /Multiplier	Direct-Method	DA
Tech.(㎛)	1.0	0.6	0.6	0.6
TRs	550K	237K	402K	378K
Throughput (Mpixels/s)	120		(16/clock)	128 (8/clock)

표 6. 전치네트워크 블록의 비교
Table 6. Comparison of transpose network block.

	Roberto Rambaldi[13]	Proposed	Notes
Transpose Network Block (TRs)	Existing RAM	Register	Efficiency
	70K	57.4K	over 18%

나. NEDA이외의 기존구조와 비교

또한 제안된 2차원 DCT/IDCT 프로세서 구조를 NEDA이외의 기존구조와 비교해 보면 표 5와 같다. 논문 [8]의 구조는 롬-기반의 구조이고, 논문 [11]의 구조는 Radix-two multi-bit coding과 곱셈기를 가지는 구조이며, 논문 [12]의 구조는 직접 방식의 구조이다. 제안한 프로세서는 기존의 롬-기반 구조[8]에 비해서는 트랜지스터 수가 172K 정도 절감하는 효과를 얻을 수 있었으며, 직접 방식[12]에 비해서는 트랜지스터 수가 약 24K정도 절감하는 효과를 얻었다. 또한 처리율에서는 롬-기반 구조[8]보다는 우월함을 확인 하였다.

표 6은 기존의 램 방식과 제안한 레지스터 적용 전치 네트워크 블록의 비교이다. 제안된 구조는 64개의 레지스터로써 구현함으로서 하드웨어의 절감을 얻을 수 있었고, 또한 램을 사용하여 설계할 때 발생되는 타이밍 제어의 어려움을 완화할 수 있었다. 따라서 논문 [13]에 비해 트랜지스터의 수를 18% 절감할 수 있었다.

V. 결 론

본 논문은 선택적으로 압축과 복원을 수행 할 수 있는 가산기 기반 DA방식의 2차원 DCT/IDCT 프로세서를 제안 설계하였다. 제안 설계된 구조는 8×8 DCT/IDCT의 계수 행렬을 4×4의 DCT/IDCT의 계수 행렬로 정규화 하여, 1차원 DCT 프로세서를 기반으로 설계되었다. 이러한 방식은 기존의 NEDA구조 방식과 비교하였을 때 48.6%의 가산기 수를 절감하였고, 새로운 알고리듬의 적용측면에서, 단지 29개의 가산기로만 계수 처리단을 수행하여 하드웨어의 면적을 크게 절감하였다. 또한 전치네트워크 블록에 있어서 기존의 방식에 비해 18%의 트랜지스터 수를 감소시켰다. 또한 기존의 다른 방식의 2차원 DCT/IDCT 프로세서에 적용이 가능하며, 이는 간단한 시스템 제어와 전치 메모리에 사용되는 클럭을 줄일 수 있는 장점이 있다.

향후 연구과제로 제안된 구조의 동작속도의 향상을 위한 개선된 연구와 하드웨어의 면적을 감소하기 위한 더욱 향상된 알고리듬의 제안이 추후로 필요하다. 또한 본 논문에서 제안 설계된 2차원 DCT/IDCT 프로세서를 기반으로 HDTV, STB, 휴대용 통신 단말기, 등의

멀티미디어 영상처리 장치에 적용, 응용될 수 있으리라 기대된다.

참 고 문 헌

[1] S.B.Pan, R.H.Park, "Two-dimentsional systolic arrays for DCT/DST/DHT hardware implemen -tation", *Journal of the Korean Institute of Telematics and Electronics* B, Vol.31B, No.10, pp.11-20, 1994.

[2] H. J. Chung, S. J. Kim, G. H. Jung, Y. D. Kim, "A DCT algorithm using shift and shift and addition", *KICS*, Vol.18, No.6, pp.773-778, June, 1993.

[3] H. S. Lim, D. S. Kim, N. I. Cho, and S. U. Lee, "Systolic Arrays for 2-D DCT and Other Orthogonal Transforms." *KITE*, Vol.27, No.7, 1990.

[4] S.W.Lee, K.B.Yim, H.J.Chung, G.H.Jung, Y.D.Kim, "A Architecture for the DCT and IDCT using a Fast DCT Algorithm", *Journal of the Korean Institute of Telematics and Electronics* B, Vol.31B, No.3, pp.11-20, 1994.

[5] Shams, A., Bayoumi, M.,"A 108 Gbps, 1.5 GHz ID-DCT architecture," *Application-Specific Systems, Architectures, and Processors, 2000. Proceedings. IEEE International Conference* on, 2000, Page(s):163 -172.

[6] Wendl Pan, Shams, A., Bayoumi, M.A.,"NEDA: a new distributed arithmetic architecture and its application to one dimensional discrete cosine transform Signal Processing Systems," 1999. *SIPS* 99. 1999 *IEEE Workshop on*, 1999, Page(s):159-168.

[7] Shams, A., Wendi Pan, Chandanandan, A.,Bayoumi , M.,"A high-performance 1D-DCT architecture," *Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. 2000 IEEE International Symposium on* Volume: 5, 2000, Page(s):521 -524. vol.5.

[8] Kim, C.S., Song, S.W., Kim, M.Y., Han, Y.T. Kang, S.A., Lee, B.W. "200 mega pixel rate IDCT processor for HDTV applications," *Circuits and Systems*, 1993., *ISCAS* '93, 1993 *IEEE International Symposium on*, May 1999, Page(s) :2003-2006 vol.3.

[9] T. S. Chang, C.S. Kung, C. W. Jen, "New distri- buted arithmetic algorithm and its application to IDCT," *Circuits and Systems for Video Techno- logy, IEEE Transactions on circuit device System* vol. 146 No. 4 Aug. 1999.

[10] B.M.Kim, H.D.Bae, T.W.Cho, "A Design of high throughput IDCT processor in Distrited Arithmetic Method", *Journal of the Institute of Electronics Engineers of Korea* SC, Vol.40, No.6, pp.48-57, 2003.

[11] Jun Rim Choi, Won Jun Hur, Kyoung Keun Lee, Ae Shin Kim, "A 400 MPixel/s IDCT for HDTV by multibit coding and group symmetry," *Solid-State Circuits Conference*, 1997. *Digest of Technical Papers*. 43rd *ISSCC.*, 1997 *IEEE International*, 1997, Page(s):262-263, 470.

[12] Yung-Pin Lee, Thou-Ho Chen, Liang-Gee Chen, Mei-Juan Chen, Chung-Wei Ku, "A cost effective architecture for 8/spl times/8 two-dimensional DCT/IDCT using direct method, Circuits and Systems for Video Technology," *IEEE Transactions* on Volume: 7. 3, June 1997, Page(s): 459-467.

[13] R. Ranbaldi, A. Ugazzoni, and R. Guerrieri, "A 35uW 1.1V gate array 8x8 IDCT processor for video-telephony," Proc. *IEEE ICASSP*, vol. 5, pp.2993-2996, 1998.

저 자 소 개

정 동 윤(정회원)
2000년 충북대학교 전자공학과 공학사.
2002년 충북대학교 전자공학과 공학석사.
2002년~현재 매그나칩 반도체
<주관심분야 : Display Driver IC, 시스템집적, 저전력회로>

서 해 준(학생회원)
2001년 청주대학교 전자·통신·반도체공학부 공학사.
2004년 충북대학교 전자공학과 공학석사.
2004년~현재 충북대학교 전자공학과 박사과정.
<주관심분야 : 시스템 집적, 저전력 회로, Display Drice IC, 반도체 메모리>

배 현 덕(평생회원)
1977년 한양대학교 전자공학과 공학사.
1980년 서울대학교 전자공학과 공학석사.
1992년 서울대학교 전자공학과 공학박사
1994년~1995년 미국 시라큐스대학 방문교수.
1987년~현재 충북대학교 전기공학과 교수
<주관심분야 : 적응 신호처리, 다중 신호처리, 웨이블릿 변환의 신호처리 응용>

조 태 원(평생회원)
1973년 서울대학교 전자공학과 공학사.
1986년 미국 루이빌대학교 전자공학과 공학석사.
1992년 미국 켄터키주립대학교 전자공학과 공학박사.
1973년~1983년 금성전선(주)
1992년~현재 충북대학교 전자공학과 교수
<주관심분야 : 시스템 집적, 고급 컴퓨터구조, 저전력회로, 반도체 메모리>