論文96-33A-11-17

ROM 방식의 곱셈기를 이용한 8×8 2차원 DCT의 구현

(The Implementation of an 8×8 2-D DCT Using ROM-based Multipliers)

李 哲 東 *, 鄭 順 基 **

(Chul-Dong Lee and Soon-Key Jung)

요 약

본 논문에서는 화상 회의, JPEG 및 MPEG에 사용 가능한 2차원 DCT의 구현에 대하여 기술한다. 구현한 DCT는 1차원 DCT 두개와 이들 사이에 전치 메모리를 사용하며, 곱셈기 대신 ROM을 사용하였다. 데이터 비트수는 C언어를 사용한 시뮬레이션을 통하여 ITU 표준 H.261에 규정된 정확도를 만족하는 최소의 수로 결정하였다. 전치 메모리로는 듀얼 포트 RAM을 이용하였으며, 덧셈 속도의 향상을 위하여 두개조의 ROM을 사용하여 입력 화소의 데이터를 한꺼번에 2비트씩 처리한다. 시스템의 기본 구조는 Synopsys사의 스키매틱 에디터를 이용하여 설계하였으며, 내부 모듈들은 VHDL로 기술하여 시뮬레이션을 거친 후에 논리 단계로 합성하는 방법을 취하였다. 논리 합성 후에는 Compass 사의 툴과 0.8um CMOS 라이브러리를 사용하여 표준 셀 방식으로 레이아웃 하였다. 최종회로는 약 11만개의 트랜지스터를 가지며, 칩의 면적은 4.68mm x 4.96mm의 크기로서, 약 50Mpixels/sec의 처리 속도를 가진다.

Abstract

This paper describes the implementation of a 2-D DCT that can be used for video conference, JPEG, and MPEG-related applications. The implemented DCT consists of two 1-D DCTs and a transposed memory between them, and uses ROM-based multipliers instead of conventional ones. As the system bit length, the minimum bit length that satisfies the accuracy specified by the ITU standard H.261 was chosen through the simulations using the C language. The proposed design uses a dual port RAM for the transposed memory, and processes two bits of input-pixel data simultaneously to speed up addition process using two sets of ROMs. The basic system architecture was designed using the Synopsys schematic editor, and internal modules were described in VHDL and synthesized to logic level after simulation. Then, the Compass silicon compiler was used to create the final layout with 0.8um CMOS libraries, using the standard cell approach. The final layout contains about 110,000 transistors and has a die area of 4.68mm x 4.96mm, and the system has the processing speed of about 50Mpixels/sec.

I. 서 론

화상신호의 압축 기술은 화상신호가 가지는 상관성을 이용하여 데이터를 압축하는 기술이다. 화상신호의 상관성에는 공간 상관성과 시간 상관성의 두 종류가 있다. 공간 상관성은 화면 내부의 인접한 화소간의 유사성을 의미한다. 시간 상관성은 현재의 화면과 전 화면에 있어서, 공간적으로 동일한 위치에 있는 화소간의 유사성을 의미한다. 정지화상의 부호화에서는 공간 상관성만을 이용하며, 동화상의 부호화에서는 공간 상관성과 시간 상관성을 이용한다.

현재까지 많은 압축기술이 제안되고 검토되어 왔다.

* 正會員, 電子部品綜合技術研究所
(Korea Electronics Technology Institute)
** 正會員, 忠北大學校 컴퓨터工學科
(Dept. of Computer Eng., Chungbuk Nat'l Univ.)
接受日字: 1996年9月13日, 수정완료일: 1996年10月30日

주요한 기본 방식으로서 예측부호화(predictive coding), 변환부호화(transform coding) 및 벡터부호화(vector coding) 등이 있다. 이와 같은 부호화 방식 중에서 변환부호화에 속하는 DCT(Discrete Cosine Transform : 이산 코사인 변환)와 움직임 보상 프레임간 예측을 조합한 방식이 국제 표준으로 채택되었다. 움직임 보상 프레임간 예측은 시간 상관성을, DCT는 공간 상관성을 이용하여 압축을 행한다.

정지화상의 부호화 방식 JPEG (Joint Photographic image coding Experts Group), 화상회의/화상전화용 부호화 방식의 ITU-T 권고 H.261[1], 축적용 부호화 방식의 MPEG1 (Moving Picture image coding Experts Group phase 1) 등이 있다. MPEG2는 통신, 방송, 저장 등의 다양한 용도에 대응하여 범용적으로 사용하는 부호화 방식을 지향하고 있다. 이상과 같이 DCT는 화상 부호화의 분야에서 필수 불가결한 기술로 되었다.

DCT에 관한 연구는 연산수를 줄이기 위한 알고리듬과 VLSI 하드웨어 아키텍쳐를 개발하는데 집중되어 왔다[2]. 본 논문에서는 두개의 1차원 DCT와 1개의 전치 행렬을 이용하여 2차원 DCT 칩을 설계하고, H.261에서 제시하는 정확도 표준을 만족하는지 확인한 후, 표준셀을 이용하여 레이아웃하고 시뮬레이션을 통하여 그 동작을 확인한다.

본 논문의 구성은 다음과 같다. II장에서는 2차원 DCT 알고리듬과 함께 전반적 구성을 기술한다. III장에서는 H.261에서 제시하는 정확도 표준화에 대한 시뮬레이션 결과를 제시하고, IV장에서는 1차원 DCT 및 전치 네크워크 구조에 대해서 설명한다. V장에서는 실험 및 고찰을 기술하고 VI장에서 결론을 맺는다.

II. DCT 알고리듬

화상은 수평과 수직 양방향으로 상관성이 있다. 따라서 화상처리에서는 일반적으로 2차원 DCT를 사용한다. 우선, 원화상을 수평방향과 수직방향으로 각각 N 화소씩 블록으로 분할하고, 분할된 N×N의 블록에 대하여 2차원 DCT를 수행한다. N이 클수록 부호화 효율이 좋지만 계산량이 증가한다. 즉, 부호화 효율과 계산량은 적절히 선택할 필요가 있다.

1. 2차원 DCT

N×N 화소 블록 $x(i,j)$ 에 대한 2차원 DCT 변환계수 $Z(u,v)$[3] 는

$$Z(u,v) = \frac{2A(u)A(v)}{N}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}x(i,j)\cos\left\{\frac{(2i+1)u\pi}{2N}\right\}\cos\left\{\frac{(2j+1)v\pi}{2N}\right\} \quad (1)$$

$$A(w)=\begin{cases}\frac{1}{\sqrt{2}} & w=0\\ 1 & w=1,2,\ldots,N-1\end{cases}$$

로 정의되고, 이를 행-열 분해(row-column decomposition)를 이용하여 행렬형태로 표현하면 다음과 같다.

$$Z = C\,x\,C^T = C\,[C\,x^T]^T \quad (2)$$

여기에서 $y = C\,x^T$ 라면, $Z = C\,[C\,x^T]^T = C\,y^T$ 가 된다. 즉 입력 x 에 대하여 1차원 DCT 수행 ($y = C\,x^T$), 행렬의 전치(y^T), 1차원 DCT 수행 ($Z = C\,y^T$)의 과정을 거침으로서 2차원 DCT의 결과를 얻을 수 있다. 두개의 1차원 DCT를 이용한 2차원 DCT를 그림 1에 보였다.

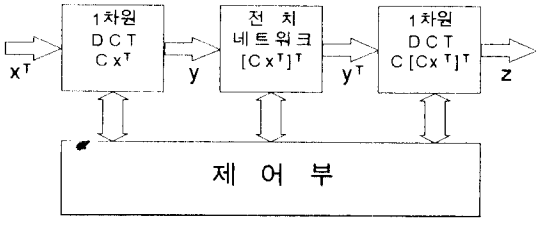


그림 1. 두개의 1차원 DCT를 이용한 2차원 DCT
Fig. 1. 2-D DCT using two 1-D DCTs.

2. 1차원 DCT

(식 2)에서 행렬 C의 기저벡터 원소 $c(k,n)$은

$$c(k,n) = \sqrt{\frac{2}{N}}A(k)\cos\frac{(2n+1)\pi k}{2N}$$

$$\text{for } n,k = 0,1,\ldots,N-1 \quad (3)$$

$$A(w)=\begin{cases}\frac{1}{\sqrt{2}} & w=0\\ 1 & w=1,2,\ldots,N-1\end{cases}$$

이고, N=8일 때 1차원 DCT $y = C\,x^T$ 를 행렬식으로 나타내면 (식 4)와 같다. 여기에서 x_i 는 입력 화소값, y_i 는 DCT 계수 값, C_k 는 코사인 계수 값을 나타낸다.

$$\begin{bmatrix}y_0\\y_1\\y_2\\y_3\\y_4\\y_5\\y_6\\y_7\end{bmatrix} = \frac{1}{2}\begin{bmatrix}C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4\\ C_1 & C_3 & C_5 & C_7 & -C_7 & -C_5 & -C_3 & -C_1\\ C_2 & C_6 & -C_6 & -C_2 & -C_2 & -C_6 & C_6 & C_2\\ C_3 & -C_7 & -C_1 & -C_5 & C_5 & C_1 & C_7 & -C_3\\ C_4 & -C_4 & -C_4 & C_4 & C_4 & -C_4 & -C_4 & C_4\\ C_5 & -C_1 & C_7 & C_3 & -C_3 & -C_7 & C_1 & -C_5\\ C_6 & -C_2 & C_2 & -C_6 & -C_6 & C_2 & -C_2 & C_6\\ C_7 & -C_5 & C_3 & -C_1 & C_1 & -C_3 & C_5 & -C_7\end{bmatrix}\begin{bmatrix}x_0\\x_1\\x_2\\x_3\\x_4\\x_5\\x_6\\x_7\end{bmatrix} \quad (4)$$

여기에서 $C_k = \cos\frac{\pi k}{16}$ 이며, 상수 $\frac{1}{2}$ 는 전체적으로 계수 값에 반영한다. 위 식을 두 개의 4x4 행렬 분해하면 (식 5)와 같다.

$$\begin{bmatrix} y_0 \\ y_2 \\ y_4 \\ y_6 \end{bmatrix} = \begin{bmatrix} C_4 & C_4 & C_4 & C_4 \\ C_2 & C_6 & -C_6 & -C_2 \\ C_4 & -C_4 & -C_4 & C_4 \\ C_6 & -C_2 & C_2 & -C_6 \end{bmatrix} \begin{bmatrix} x_0+x_7 \\ x_1+x_6 \\ x_2+x_5 \\ x_3+x_4 \end{bmatrix} \tag{5a}$$

$$\begin{bmatrix} y_1 \\ y_3 \\ y_5 \\ y_7 \end{bmatrix} = \begin{bmatrix} C_4 & -C_4 & -C_4 & C_4 \\ -C_3 & -C_7 & C_1 & -C_5 \\ -C_6 & C_2 & -C_2 & C_6 \\ C_1 & -C_3 & C_5 & -C_7 \end{bmatrix} \begin{bmatrix} x_0-x_7 \\ x_1-x_6 \\ x_2-x_5 \\ x_3-x_4 \end{bmatrix} \tag{5b}$$

(식 5)에서 입력 화소 값의 합과 차에 대한 식을 $u_0 = x_0+x_7$, $u_1 = x_1+x_6$, $u_2 = x_2+x_5$, $u_3 = x_3+x_4$, $v_0 = x_0-x_7$, $v_1 = x_1-x_6$, $v_2 = x_2-x_5$, $v_3 = x_3-x_4$ 로 치환하면 1차원 DCT는 다음과 같다.

$$\begin{bmatrix} y_0 \\ y_2 \\ y_4 \\ y_6 \end{bmatrix} = \begin{bmatrix} C_4 & C_4 & C_4 & C_4 \\ C_2 & C_6 & -C_6 & -C_2 \\ C_4 & -C_4 & -C_4 & C_4 \\ C_6 & -C_2 & C_2 & -C_6 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{bmatrix} \tag{6a}$$

$$\begin{bmatrix} y_1 \\ y_3 \\ y_5 \\ y_7 \end{bmatrix} = \begin{bmatrix} C_4 & -C_4 & -C_4 & C_4 \\ -C_3 & -C_7 & C_1 & -C_5 \\ -C_6 & C_2 & -C_2 & C_6 \\ C_1 & -C_3 & C_5 & -C_7 \end{bmatrix} \begin{bmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \end{bmatrix} \tag{6b}$$

3. 분산산술 (distributed arithmetic)

DCT 회로의 핵심인 1차원 DCT는 분산산술 (distributed arithmetic)[4][5][6]을 이용한다. 분산산술은 벡터와 행렬의 곱셈을 수행할 때 벡터를 비트 단위로 순차적으로 처리하는 방법이다. 동일한 차수를 가지는 비트들의 모든 비트 패턴에 대한 부분곱 (partial product)을 ROM에 저장해 놓고, 이 부분곱을 비트의 차수를 고려하여 계속적으로 더하여 곱셈 결과를 얻는다. ROM과 가산기가 연결되어 있는 것을 RAC(ROM and Accumulator in Cascade)라고 하며 이는 분산산술 처리 방식의 기본이 되는 요소이다.

N×N 입력화소 행렬 X 의 1차원 DCT 를 $Y = CX$ 로 다시 정의하자. 그러면 Y의 i 번째 열과 k 번째 행의 값은 다음과 같다.

$$y_{ik} = \sum_{m=0}^{N-1} c_{mk}\, x_{im} \quad i,k = 0,1,2,\ldots,N-1 \tag{7}$$

또 x_{im} 를 p-비트 2의 보수 형태로 표현하면 다음과 같다.

$$x_{im} = -x_{im}^{(p-1)}\, 2^{p-1} + \sum_{q=0}^{p-2} x_{im}^{(q)}\, 2^q \tag{8}$$

여기에서 $x_{im}^{(q)}$ 는 x_{im} 의 q 번째 비트이며 그 값은 0 혹은 1이다. 2^q 는 q 번째 비트의 2진 가중치이다. (식 8)를 (식 7)에 대입하면 다음과 같다.

$$y_{ik} = -\sum_{m=0}^{N-1} c_{mk}\, x_{im}^{(p-1)}\, 2^{p-1} + \sum_{q=0}^{p-2}\sum_{m=0}^{N-1} c_{mk}\, x_{im}^{(q)}\, 2^q \tag{9}$$

(식 9)를 다시 정의하여,

$$y_{ik} = -F_{ik}(C_k, X_i^{(p-1)})\, 2^{p-1} + \sum_{q=0}^{p-2} F_{ik}(C_k, X_i^{(q)})\, 2^q \tag{10}$$

로 표현하면 F_{ik} 는 벡터 $C_k, X_i^{(q)}$ 의 함수로써 $q = 0,1,2,\ldots,p-1$ 일 때

$$F_{ik}(C_k, X_i^{(q)}) = \sum_{m=0}^{N-1} c_{mk} x_{im}^{(q)} \tag{11a}$$

$$F_{ik}(C_k, X_i^{(q)}) = c_{0k}x_{i0}^{(q)} + c_{1k}x_{i1}^{(q)} \ldots + c_{N-1k}x_{iN-1}^{(q)} \tag{11b}$$

로 표현된다. 여기에서 c_{mk} 는 기저벡터로서 십진수이고, $x_{im}^{(q)}$ 은 동일한 가중치를 갖는 N개의 비트 패턴이다. 따라서 F_{ik} 는 기저 벡터와 비트 패턴의 함수이다. 기저벡터는 상수이므로 (식 11)의 F_{ik} 를 구할 때는 2^q 의 가중치를 가지는 N개($m = 0,1,\ldots,N-1$)의 $x_{im}^{(q)}$ 의 모든 비트 패턴에 대하여 값을 계산하여 이를 메모리에 저장한다. (식 10)으로 부터 y_{ik} 는 메모리에 저장되어 있는 F_{ik}를 쉬프트와 더하기를 수행함으로서 얻을 수 있다. 또한 (식 11)에서 F_{ik}는 각 k의 값에 따라서 서로 다르다. 여기에서 k는 기저벡터 C_k 와 관계된다. $k = 0,1,\ldots,N-1$ 의 N개의 분리된 메모리 뱅크를 가짐으로서 N개의 1차원 DCT 계수 y_{ik} ($k = 0,1,\ldots,N-1$) 는 병렬 계산이 가능하다. N=8일 때 입력벡터는 8개의 요소를 가지며 동일한 가중치를 가지는 비트는 각 요소마다 1개씩 있다. 따라서 동일한 가중치를 가지는 8개의 비트로 표현 가능한 패턴은 256 가지이다. 따라서 각 DCT 계수에 대하여 256 워드 ROM이 필요하다.

DCT 기저 벡터는 코사인 주기성에 의하여 다음과 같은 대칭성을 가진다.

$$c_{mk} = c_{N-1-m,k} \quad \text{for} \quad k = 0,2,\ldots N-2 \tag{12a}$$

$$c_{mk} = -c_{N-1-m,k} \quad \text{for } k = 1,3,\ldots N-1 \tag{12b}$$

따라서 (식 7)은 다음과 같이 표현할 수 있고,

$$y_{ik} = \sum_{m=0}^{N/2-1}(x_{im}+x_{i,N-1-m})\ c_{mk}$$
$$k=0,2,\ldots N-2 \qquad (13a)$$

$$y_{ik} = \sum_{m=0}^{N/2-1}(x_{im}-x_{i,N-1-m})\ c_{mk}$$
$$k=1,3,\ldots N-1 \qquad (13b)$$

(식 11)은 다음과 같이 유도된다.

$$F_{ik}(C_k, X_i^{(q)}) = \sum_{m=0}^{N/2-1} c_{mk}\ (x_{im}+x_{i,N-1-m})^{(q)}$$
$$k=0,2,\ldots N-2 \qquad (14a)$$

$$F_{ik}(C_k, X_i^{(q)}) = \sum_{m=0}^{N/2-1} c_{mk}\ (x_{im}-x_{i,N-1-m})^{(q)}$$
$$k=1,3,\ldots N-1 \qquad (14b)$$

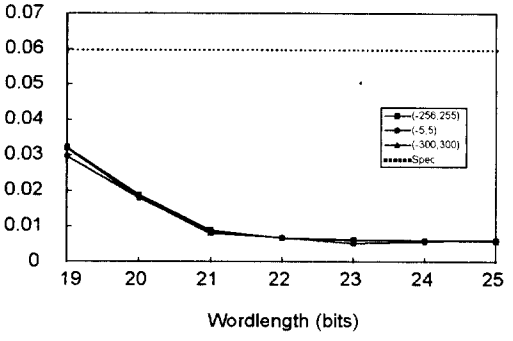
(식 14)의 F_{ik} 를 구할 때는 2^q의 가중치를 가지는 N/2개($m=0,1,\ldots,N/2-1$)의 $x_{im}^{(q)}$의 모든 가능한 비트 패턴에 대하여 값을 계산하여 메모리에 저장한다. 이 방법에서는 F_{ik}를 구하기 위해 데이터의 합과 차를 먼저 계산한다. 이렇게 함으로써 2^q의 가중치를 갖는 비트의 수를 N에서 N/2로 줄인다. 본 논문에서 N=8 이므로 원래는 각 DCT 계수에 대해서 256 워드 ROM이 1개 필요하였으나, 상기 방법에 의해 16 워드 ROM 1개와 가산기 또는 감산기가 각각 1개씩 필요하다.

Ⅲ. DCT/IDCT 정확도

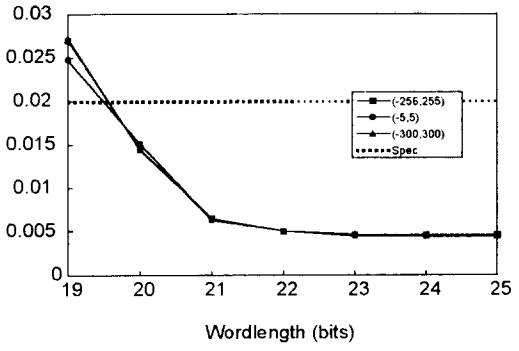
ITU는 H.261에서 IDCT의 정확도에 대한 규정을 두고 있으며 이를 간략하게 설명하면 다음과 같다.

1) H.261의 난수발생 프로그램을 이용하여 -L 과 +H 사이의 값을 갖는 정수의 난수를 발생시켜서 8×8 블록으로 정렬한다. (L=256, H=255), (L=H=5) 및 (L=H=300) 에 대하여 각각 10,000개의 데이터 세트를 준비한다.
2) 각 8×8 블록에 대하여 64비트 부동소수점 연산을 사용하여 DCT를 수행하여 결과치를 반올림하여 정수로 한다. 이때 결과치는 -2048 과 +2047 사이의 값이 되도록 한다. 이는 IDCT의 12비트 입력 데이터가 된다.
3) 2)에서 주어진 각 8×8 블록에 대하여 64비트 부동소수점 연산 방법으로 IDCT를 수행한 후, 결과치를 반올림하여 -256과 +255 사이의 정수 값이 되도록 한다.
4) 2)에서 주어진 각 8 x 8 블록에 대하여 제안한 방법으로 IDCT를 수행한 후 결과치를 반올림하여 -256 과 +255 사이의 정수 값이 되도록 한다.
5) 1)에서 구한 10,000개의 데이터 세트에 대하여 2), 3) 및 4)의 과정을 반복하고, 3)과 4)의 결과치에 대하여 아래의 사양을 만족하여야 한다.
   - o 임의의 화소에 대하여 pel peak error의 값은 1을 넘지 않아야 한다.
   - o 임의의 화소에 대하여 pel mean square error는 0.06을 넘지 않아야 한다.
   - o Overall mean square error는 0.02를 넘지 않아야 한다.
   - o 임의의 화소에 대하여 pel mean error는 0.015를 넘지 않아야 한다.
   - o Overall mean error는 0.0015를 넘지 않아야 한다.

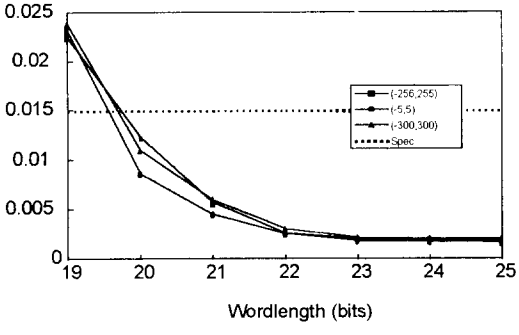
DCT와 IDCT의 내부 구조는 동일하다. 따라서 DCT의 데이터패스 구조를 결정하기 위하여 IDCT의 정확도를 만족하는 데이터패스의 비트수를 구하여 DCT에 적용하고자 한다. DCT의 VLSI 구조는 이미 많이 알려져 있다. DCT 설계시 가장 중요한 고려 사항 중의 하나는 데이터패스의 비트수를 결정하는 것이다. 본 논문에서는 데이터패스의 비트수를 결정하기 위하여 IDCT를 C 언어로 구현하여 시뮬레이션을 수행하였다. 시뮬레이션은 데이터패스의 비트수를 가변으로 하여 앞에서 설명한 5가지 조건에 대하여 실시하였다. 임의의 화소에 대한 pel peak error는 1을 넘지 않았으며, 4 종류의 오차를 그림 2에 나타내었다. 그림 2에서 주어진 오차를 만족하는 비트는 20비트임을 알 수 있다. 따라서 DCT 데이터패스의 비트수는 20비트로 결정하였다.



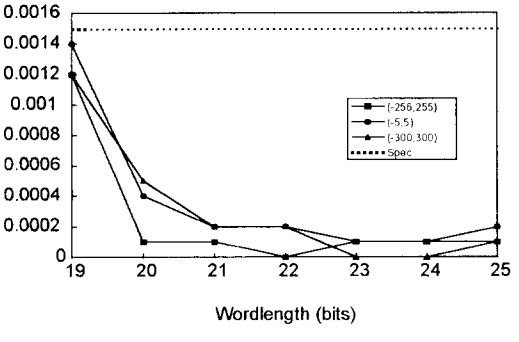
(a) Pel Mean Square Error



(b) Overall Mean Square Error



(c) Pel Mean Error



(d) Overall Mean Error

그림 2. 비트수 결정을 위한 시뮬레이션
Fig. 2. Simulation for choosing the word length.

Ⅳ. DCT 구조

1. 1차원 DCT

그림 3에 1차원 DCT[7]의 구조를 나타내었다. 1차원 DCT의 구조를 각 단계별로 설명하면 다음과 같다.

1) PREQ1: 1개의 병렬 쉬프트 레지스터와 두개의 시리얼 쉬프트 레지스터로 구성된 입력단이다. 8×8 블록 입력 벡터는 PREQ1 내부의 병렬 쉬프트 레지스터로 매 클럭마다 하나씩 로드됨과 동시에 로드된 입력 데이터는 다음 단으로 쉬프트되어, 8클럭 사이클 안에 모든 PREQ1의 병렬 쉬프트 레지스터가 데이터를 가진다. 9번째 클럭에서 내부의 병렬 쉬프트 레지스터의 내용은 홀수 비트와 짝수 비트로 나누어져 두개의 시리얼 쉬프트 레지스터로 로드된다. 시리얼 쉬프트 레지스터에 로드된 데이터는 매 클럭마다 1비트씩 쉬프트되어 FS2BIT 블록 및 FA2BIT 블록에 입력된다.

2) FS2BIT: FS2BIT는 (식 6b)의 $v_0=x_0-x_7$, $v_1=x_1-x_6$, $v_2=x_2-x_5$, $v_3=x_3-x_4$ 을 구하기 위한 2비트 감산기로써 2비트씩 감산하여 v_0, v_1, v_2, v_3를 구한다.

3) FA2BIT: FA2BIT는 (식 6a)의 $u_0=x_0+x_7$, $u_1=x_1+x_6$, $u_2=x_2+x_5$, $u_3=x_3+x_4$을 구하기 위한 2비트 가산기로써 2비트씩 가산하여 u_0, u_1, u_2, u_3를 구한다.

4) ROM: (식 14)의 $F_{ik}(C_k, X_i^{(q)})$를 계산하여 보관하는 기억소자이다.

5) ADDSUB: (식 10)의 부분곱을 구하는 20비트 가산기이다. 최상위 비트에 대해서는 감산을 수행하고 나머지 비트에 대해서는 가산을 수행한다.

6) FAACC1: (식 10)의 부분곱을 전부 더하는 누산기이다. y_{ik}를 계산하는데 5 클럭 사이클이 필요하다. FAACC1의 최종결과는 9번째 클럭에 출력 버퍼 QOUT1에 로드됨과 동시에 누산기 FAACC1은 다음 사이클을 위하여 클리어 된다.

7) QOUT1: FAACC1의 결과는 9번째 클럭에 병렬로 출력 버퍼 QOUT1에 로드된다. 출력 버퍼에 로드된 1차원 DCT 결과는 8클럭 동안 한 계수씩 순차적으로 출력되어 전치 메모리에 저장된다.

2. ROM

(식 6a)에서 u_0, u_1, u_2, u_3가 1비트 데이터라면 y_0, y_2, y_4, y_6 는 각각 16 종류의 데이터를 가지는데, 이를 미리 계산하여 ROM0, ROM1, ROM2, ROM3에 각각 저장한다. 같은 방법으로 (식 6b)의 y_1, y_3, y_5, y_7에 대한 값을 계산하여 ROM4, ROM5, ROM6, ROM7에 저장한다. ROM의 값은 표 1과 같다.

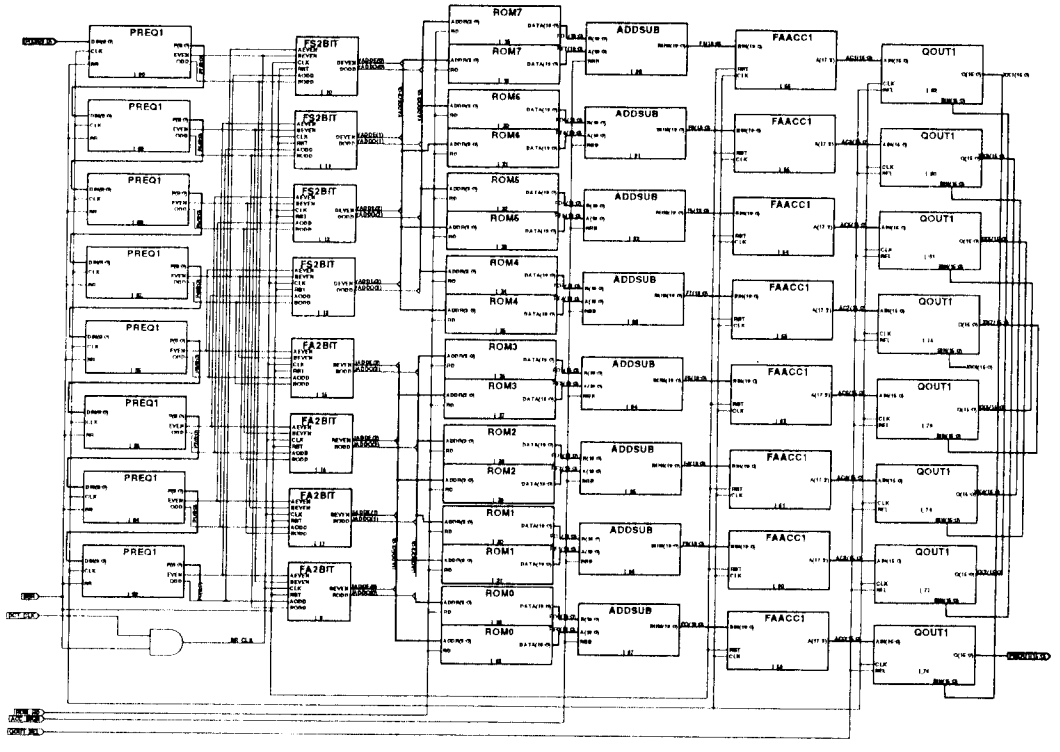


그림 3. 1차원 DCT 구조
Fig. 3. Block diagram of 1-D DCT.

표 1. 분산산술을 위한 ROM 데이터
Table 1. ROM data for distributed arithmetic.

번지	ROM 내용				번지	ROM 내용			
u_3u_0	ROM 0	ROM1	ROM 2	ROM3	v_3v_0	ROM4	ROM5	ROM6	ROM7
0000	0	0	0	0	0000	0	0	0	0
0001	C_4	C_2	C_4	C_6	0001	C_1	C_3	C_5	C_7
0010	C_4	C_6	$-C_4$	$-C_2$	0010	C_3	$-C_7$	$-C_1$	$-C_5$
0011	$2C_4$	C_2+C_6	0	C_6-C_2	0011	C_1+C_3	C_3-C_7	$-C_1-C_5$	$-C_5+C_7$
0100	C_4	$-C_6$	$-C_4$	C_2	0100	C_5	$-C_1$	C_7	C_3
0101	$2C_4$	C_2-C_6	0	C_2+C_6	0101	C_1-C_5	$-C_1+C_3$	C_5+C_7	C_3+C_7
0110	$2C_4$	0	$-2C_4$	0	0110	C_3+C_5	$-C_1-C_7$	$-C_1-C_7$	C_3-C_5
0111	$3C_4$	C_2	$-C_4$	C_6	0111	$C_1+C_3+C_5$	$-C_1+C_3-C_7$	$-C_1-C_5+C_7$	$C_3-C_5+C_7$
1000	C_4	$-C_2$	C_4	$-C_6$	1000	C_7	$-C_5$	C_3	$-C_1$
1001	$2C_4$	0	$2C_4$	0	1001	C_1+C_7	C_3-C_5	C_3+C_5	$-C_1+C_7$
1010	$2C_4$	$-C_2+C_6$	0	$-C_2-C_6$	1010	C_3+C_7	$-C_5-C_7$	$-C_1+C_3$	$-C_1-C_5$
1011	$3C_4$	C_6	C_4	$-C_2$	1011	$C_1+C_3+C_7$	$C_3-C_5-C_7$	$-C_1+C_3+C_5$	$-C_1-C_5+C_7$
1100	$2C_4$	$-C_2-C_6$	0	C_2-C_6	1100	C_5+C_7	$-C_1-C_5$	C_3+C_7	$-C_1+C_3$
1101	$3C_4$	$-C_6$	C_4	C_2	1101	$C_1+C_5+C_7$	$-C_1+C_3-C_5$	$C_3+C_5+C_7$	$-C_1+C_3+C_7$
1110	$3C_4$	$-C_2$	$-C_4$	$-C_6$	1110	$C_3+C_5+C_7$	$-C_1-C_5-C_7$	$-C_1+C_3+C_7$	$-C_1+C_3-C_5$
1111	$4C_4$	0	0	0	1111	$C_1+C_3+C_5+C_7$	$-C_1+C_3-C_5-C_7$	$-C_1+C_3+C_5-C_7$	$-C_1+C_3-C_5-C_7$

주) $C_k = \cos\frac{\pi k}{16}$

3. 전치 네트워크

그림 4에 전치 네트워크의 구조를 보였다. 전치 메모리 TRAM(Transpose RAM)은 듀얼 포트 RAM을 이용하였다.

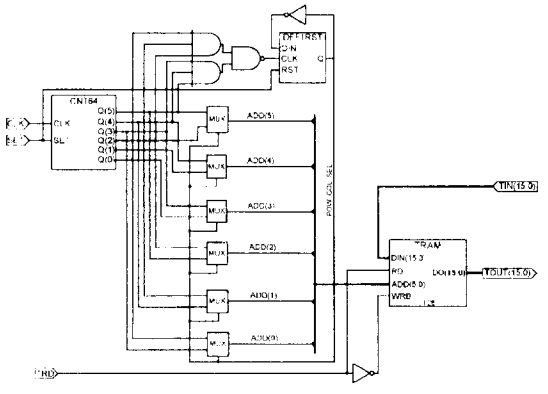


그림 4. 듀얼 포트 RAM을 사용하는 전치 네트워크
Fig. 4. Transposition network using a dual port RAM.

TRAM은 8×8 계수의 행과 열을 교환하는데 필요하다. 첫 번째 1차원 DCT에 의하여 얻어진 계수의 새로운 8×8 블록은 TRAM에 저장된다. 동시에 먼저 저장된 8×8 블록의 계수는 전치(transpose)된 순서로 읽혀진다. 따라서 RAM은 동일한 사이클에 읽기와 저장이 이루어져야 한다. 계수는 TRAM에 순차적으로 쓰여진다. 즉 8×8 블록의 첫 번째 행의 계수 0,1,2,...,7의 순서로 먼저 저장되고, 다음에 두 번째 행의 0,1,2...,7의 순서로 저장되어 8번째 행까지 진행한다. 이는 8×8 블록의 계수 0,1,2,...63의 긴 스트림으로 생각할 수 있다. 이렇게 입력된 계수는 전치 기능을 얻기 위하여 8의 오프셋을 가지고 (0, 8, 16, ... 56) (1, 9, 17, ..., 57) (2, 10, 18, ..., 58) (3, 11, 19, ..., 59) (4, 12, 20, ..., 60) (5, 13, 21, ..., 61) (6, 14, 22, ..., 62) (7, 15, 23, ..., 63)의 순으로 읽혀진다. 첫 번째 블록의 계수가 읽혀지고, 두 번째 8×8 블록 계수가 동일한 위치에 저장된다. 이제 블록 2의 계수가 전치 기능을 위하여 0,1,2,...,63의 순차적인 순서로 읽혀진다. 동시에 블록 3의 계수가 블록 1에서와 같은 순서, 즉 블록 2의 0을 읽고 블록 3의 0을 저장하고, 블록 2의 1을 읽고 블록 3의 1을 저장하고, 블록 2의 2를 읽고 블록 3의 2를 저장하는 순서로 저장한다. 각 클럭 사이클에 읽고 저장하는 과정이 처음에 0,1,2...,63의 순차적인 순서로, 다음에 오프셋 순서인 0,8,16,... 으로, 다시 0,1,2...,의 순차적인 순서로 계속된다.

V. 실험 및 고찰

두개의 1차원 DCT와 전치 네트워크로 구성된 2차원 DCT의 유용성을 검증하기 위하여 Synopsys사와 Compass사의 툴을 이용하여 2차원 DCT를 구현하였다.

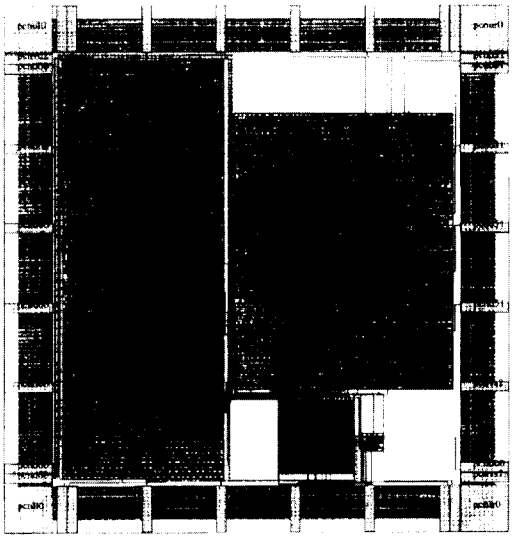


그림 5. 최종 레이아웃
Fig. 5. The final layout.

그림 3의 회로는 Synopsys의 스키매틱 에디터를 이용하여 설계한 1차원 DCT의 최상위 레벨이다. 먼저, 각 모듈의 동작을 VHDL로 기술한 후, 시뮬레이션을 통하여 확인하였다. 동작을 확인한 후에는 Synopsys 툴을 이용하여 자동으로 논리합성 하였다. 전치 네트워크는 Compass사에서 제공하는 라이브러리인 듀얼 포트 RAM에 주변회로를 첨가하여 구성하였다. 1차원 DCT와 전치 네트워크를 결합하여 Compass 툴에서 레이아웃을 수행하였으며 그림 5에 레이아웃 결과를 보였다. 레이아웃은 CMOS 0.8um 선폭의 표준셀 방식으로 수행하였으며, 듀얼 포트 RAM등은 Compass사에서 제공하는 라이브러리를 이용하였다. 칩 크기는 4.68mm x 4.96mm, 코아 크기는 3.64mm x 3.94mm이며, 사용된 트랜지스터는 약 11만개이다. DCT 회로에서 속도에 가장 큰 영향을 미치는 부분은 그림 3의

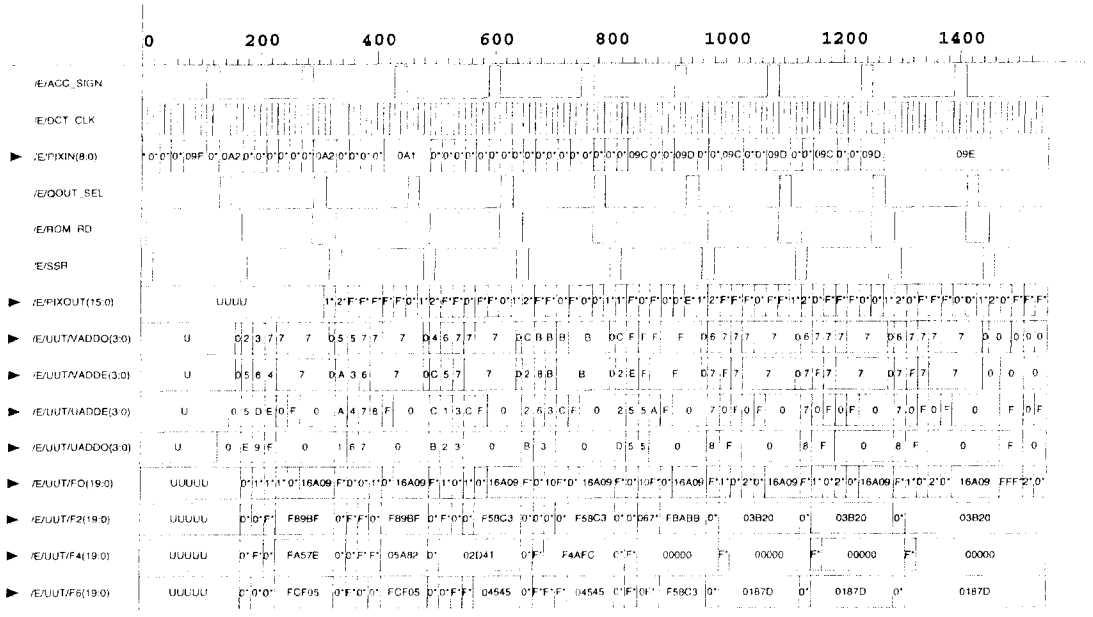


그림 6. 시뮬레이션 결과
Fig. 6. Simulation result.

ADDSUB 및 FAACC1의 20비트 가산기 회로이다. 덧셈 실행속도를 높이기 위하여 예견 올림자리수(look-ahead carry)에 의한 가산기를 합성에 의하여 설계하였다.

그림 6의 시뮬레이션 결과는 1차원 DCT의 시뮬레이션 결과이다. 시뮬레이션의 입력 데이터는 H.261의 난수발생 프로그램을 이용하여 생성한 랜덤 데이터이며, 출력은 1차원 DCT 후의 값이다. 데이터가 입력되기 시작한 후, 16 클럭이 지나면서부터 1차원 DCT의 결과가 출력된다. 또, 클럭의 주기가 20 ns이므로, 클럭 주파수는 50MHz이다. 한 클럭당 1개의 픽셀이 처리되므로 설계된 2차원 DCT의 처리속도는 50Mpixels/sec이다.

그림 7은 2차원 DCT의 타이밍도이다. 8×8 블록의 픽셀 데이터를 쉬프트 레지스터 PREQ1에 순차적으로 로드하는데 8클럭, 1차원 DCT를 수행하는데 8클럭이 필요하다. 즉, 첫 번째 픽셀이 입력된 후 첫 번째 1차원 DCT의 결과를 16클럭부터 얻게 된다. 첫 번째 1차원 DCT의 결과는 전치 네트워크에 저장된다. 64개의 결과를 저장해야 하므로 64 클럭이 필요하다. 또, 두 번째 1차원 DCT를 수행하는데 16클럭이 필요하다. 따라서, 픽셀 데이터가 입력된 후, 총 96클럭 후에 2차원 DCT의 출력이 생성된다.

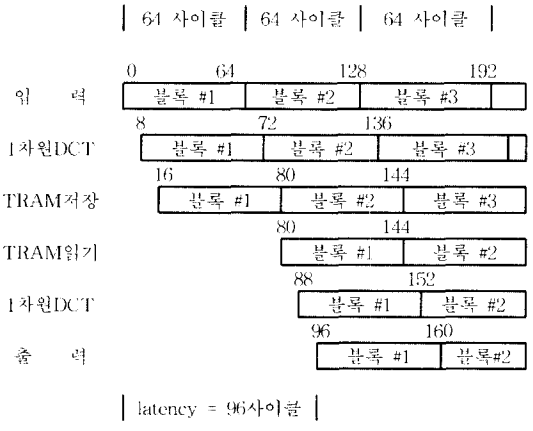


그림 7. 2차원 DCT의 데이터 타이밍
Fig. 7. Data timing of 2-D DCT.

표 2는 기존의 DCT 또는 IDCT 칩들과 본 논문에서 구현한 DCT 칩의 데이터 패스의 비트수를 비교한 것이다. 표 2에 제시한 칩들은 모두 행-열 분해 방식에 근거한 DCT 또는 IDCT 칩으로서 H.261 표준에서

제시하는 정확도를 모두 만족한다. 행-열 분해방식에 의한 DCT 또는 IDCT에서 데이터패스의 비트수가 작을수록 데이터 처리시간이 짧다. 그런데, 표 2에서 알 수 있는 바와 같이 본 논문의 DCT 칩은 20비트의 데이터패스를 가진다. 따라서, 본 논문의 DCT 설계가 기존의 DCT 또는 IDCT 설계보다 우수하다는 것을 알 수 있다.

표 2. 데이터패스의 비트수 비교
Table 2. Comparison of the number of datapath.

방법	[3]	[8]	[9]	본 논문
비트수	22	25	32	20

표 3은 본 논문에서 설계한 DCT 칩의 특성을 요약한 것이다. 코아의 크기는 14.3 mm^2이며 약 50 MHz에서 동작한다.

표 3. 칩 특성
Table 3. Chip characteristics.

입력	9 비트
출력	12 비트
칩 크기	23.2 mm^2
코아 크기	14.3 mm^2
테크놀로지	0.8 um CMOS
트랜지스터	110,000 개
클럭	50 MHz
데이터패스	20 비트
처리속도	50 Mpixels/sec
정확도	H.261 표준 만족

VI. 결 론

본 논문에서는 두개의 1차원 DCT와 1개의 전치 행렬을 이용하여 2차원 DCT 칩을 설계했다. 첫째, ITU의 권고 H.261에서 제시하는 정확도 표준을 만족하는 데이터패스의 비트수를 결정하기 위하여 C언어로 프로그램을 작성하여 시뮬레이션을 수행하였다. 둘째, 시뮬레이션을 통하여 결정한 비트수의 데이터패스를 가지는 1차원 DCT의 각 모듈을VHDL을 이용하여 설계하였다. 1차원 DCT의 구조는 곱셈기 대신 ROM과 가산기를 이용하였다. 또한, 전치 네트워크는 듀얼 포트 RAM을 사용하여 RAM을 읽으면서 저장하기 편하도록 설계하였다. 셋째, 1차원 DCT와 전치 네트워크를 결합하여 2차원 DCT를 설계할 때는 스키매틱 에디터를 이용하여 결합하였고 Synopsys 툴을 이용하여 자동합성하였다. 넷째, 합성된 네트리스트는 Compass 툴을 이용하여 레이아웃을 수행하였고, 설계된 최종 레이아웃은 시뮬레이션을 통하여 검증하였다.

본 논문의 2차원 DCT는 두개의 1차원 DCT와 전치 네트워크로 구성되어 있고, 50Mpixels/sec의 처리속도 및 4.68mm x 4.96mm의 크기를 가진다. 본 논문 DCT 칩은 화상회의, JPEG 및 MPEG의 DCT에 사용 가능하다.

참 고 문 헌

[1] ITU, ITU T Recommendation H.261, 1993.

[2] S. Hsia, D. Liu, J. Yang, and B. Bai, "VLSI implementation of parallel coefficient -by coefficient two-dimensional IDCT processor," *IEEE Trans. Circuit and Systems for Video Technology*, vol. 5, no. 5, pp. 396-406, Oct. 1995.

[3] A. Madisetti and A. N. Wilson, "A 100MHz 2 D 8x8 DCT/IDCT processor for HDTV applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 2, pp. 158-165, Apr. 1995.

[4] W. Chen, Y. Fan, and K. W. Hsu, "An 8×8 discrete cosine transform model using VHDL," *Proc. 4th ASIC Conference and Exhibit*, pp. 339-344, 1992

[5] T. Sun, L. Wu, and M. L. Liou, "A concurrent architecture for VLSI implementation of discrete cosine transform," *IEEE Trans. Circuits and Systems*, vol. CAS 34, no. 8, pp. 992-994, Apr. 1987.

[6] S. Uramoto, Y. Inouc, A. Takabatake, J. Takeda, Y. Yamashita, H. Terane, and M. Yoshimoto, "A 100MHz 2-D discrete cosine transform core processor," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 4, pp. 492 498, Apr. 1992.

[7] L. J. D'Luna et al., "An 8×8 discrete cosine

transform chip with pixel rate clocks," *Proc. 3rd ASIC Conference and Exhibit*, pp. P7-5.1-4, 1990.

[8] C. S. Kim, S. W. Song, M. Y. Kim, Y. T. Han, S. A. Kang, and B. W. Lee, "200 mega pixel rate IDCT processor for HDTV applications," *IEEE ISCAS*, pp. 2003-2006, 1993.

[9] 임근호, 류근장, 권용무, 김형곤, "완전 비트 순차 구조에 근거한 2차원 DCT/IDCT VLSI 구현", 전자공학회논문지, 제31권 A편 제6호, pp. 188-198, 1996년 6월.

저 자 소 개

李 哲 東(正會員)

1952년생. 경북대학교 전자공학과 학사. 한양대학교 전자공학과 석사. 한국전자기술연구소 설계자동화연구실 실장(1977-). 한국전자통신연구소 자동설계연구부 부장(1985-). 전자부품종합기술연구소 주문형반도체설계센터 센터장(1994-현재). 전자공학회 CAD 및 VLSI 설계연구회 전문위원장(1995-현재). 주 관심분야는 집적회로 설계, 반도체용 자동설계툴 개발 등임.

鄭 順 基(正會員)

1944년생. 고려대학교 학사. 독일 Dortmund대학교 전산학과 석사(Dipl.-Inf.). 네델란드 Groningen 대학교 전산학과 박사(Dr.). 독일 Philips Data Systems 근무(1983). 데이타통신 정보통신연구소 근무(1985). 충북대학교 컴퓨터공학과 교수(1985.5 - 현재). 충북대학교 전자계산소장(1994). 주 관심분야는 실시간 시스템, DBS, 소프트웨어공학 등임