論文95-32A-1-27

Vector-radix 2차원 고속 DCT의 VLSI 어레이 구현

(A VLSI Array Implementation of Vector-radix 2-D Fast DCT)

姜 龍 暹 **, 全 興 雨 *, 辛 卿 旭 *

(Yong-Seom Kang, Heung-Woo Jeon, and Kyung-Wook Shin)

요 약

이산 코사인 변환 (discrete cosine transform: DCT) 의 고속 알고리듬인 vector-radix 2차원 DCT (VR-FCT) 를 병렬계산하기 위한 어레이 회로를 설계하였다. VR-FCT 알고리듬의 버터플라이 연산부분을 2차원 어레이에 매핑하여 병렬 및 파이프라인 처리함으로써 VR-FCT 알고리듬의 고속성과 2차원 어레이의 병렬성 및 국부통신 특성을 동시에 이용한다. 제안된 구현방식은 행-열 분해방식과는 달리 transposition 메모리가 필요치 않으며, 2차원 어레이의 구조적인 규칙성, 모듈성 및 국부연결성 등에 의해 VLSI 구현에 매우 적합하다는 특징을 갖는다. 반도체공동연구소(ISRC)의 1.5μm N-Well CMOS 공정을 이용하여 설계된 (8 x 8) 2차원 DCT 계산용 어레이 코어는 64개의 처리요소가 (8 x 8) 2차원 배열로 구성되며, 약 98,000여개의 트랜지스터와 138mm^{2}의 면적을 갖는다. 시뮬레이션 결과에 의하면, 50 MHz 클럭주파수에서 (8 x 8) 2차원 DCT 계산에 약 0.88 μsec가 소요되며, 약 72x10^6 pixels/sec의 연산성능이 예상된다.

Abstract

An array circuit is designed for parallel computation of vector-radix 2-D discrete cosine transform (VR-FCT) which is a fast algorithm of DCT. By using a 2-D array of processing elements (PEs), the butterfly structure of the VR-FCT can be efficiently implemented with high concurrency and local communication geometry. The proposed implementation features architectural modularity, regularity and locality, so that it is very suitable for VLSI realization. Also, no transposition memory is required. The array core for (8 x 8) 2-D DCT, which is designed using ISRC 1.5μm N-Well CMOS technology, consists of 64 PEs arranged in (8 x 8) 2-D array and contains about 98,000 transistors on an area of 138mm^2. From simulation results, it is estimated that (8 x 8) 2-D DCT can be computed in about 0.88 μsec at 50 MHz clock frequency, resulting in the throughput rate of about 72x10^6 pixels per second.

* 正會員, 金烏工科大學校 電子工學科

(Dept. of Elec. Eng., Kumoh Nat'l Univ.)

** 正會員, 高麗大學校-KIST學硏

接受日字 : 1994年 3月 25日

I. 서 론

영상신호의 압축을 위해 가장 일반적으로 사용되는 방법인 이산 코사인 변환 (discrete cosine transform: DCT) 은 JPEG, MPEG, CCITT H.261과 같은 국제 표준규격에서 영상압축 방식의 표준 알고리듬

으로 채택되고 있으며, 특히 디지탈 HDTV 방식에서도 DCT를 기본으로한 대역폭 압축방법이 연구되고 있다.[1-4] 최근에 HDTV, G4 FAX 및 CD-I (Compact Disk-Interactive) 등 각종 멀티미디어 시스템들의 실용화가 이루어지면서 VLSI (very large scale integration) 기술을 이용한 고성능 2차원 DCT 프로세서의 개발에 대한 중요성이 더욱 증대되고 있다. 실시간 영상처리를 위한 고성능 2차원 DCT 프로세서의 개발을 위해서는 규칙적인 연산구조를 갖는 고속 알고리듬의 선택, 국부통신구조를 갖는 아키텍처의 설계, 그리고 알고리듬을 아키텍처에 효율적으로 매핑할 수 있는 매핑 알고리듬의 개발 등이 중요한 요소가 된다.[17,20]

DCT 프로세서의 구현방법은 크게 나누어 2차원 DCT의 분해특성을 이용하는 Row-Column Algorithm(RCA) 방식과 분해특성을 이용하지 않는 Non Row-Column Algorithm (NRCA) 방식으로 구분할 수 있다.[5] RCA 방식[6-11]은 2차원 영상에 대해 행방향 (또는 열방향)의 1차원 DCT를 수행한 후, 그 결과를 matrix transposition 하여 열방향 (또는 행방향)의 1차원 DCT를 수행함으로써 2차원 DCT를 계산한다. 이 방식은 1차원 DCT 구조를 이용하여 2차원 DCT를 계산할 수 있다는 장점이 있으나, transposition 메모리가 부가적으로 필요하다는 단점이 있다. 한편, vector-radix 분해[12,13], polynomial 변환[14], 2-D DCT의 1-D DCT 분해[15] 등을 이용하는 고속 알고리듬, Winograd 변환을 이용한 알고리듬[16] 등은 NRCA 범주에 속하며, matrix transposition 메모리가 필요치 않다는 장점을 갖는다.

본 논문에서는 vector-radix 2차원 고속 DCT (VR-FCT) 알고리듬[13]의 효율적인 VLSI 구현방법을 제안하고, 이를 집적회로로 설계하였다. VR-FCT 알고리듬은 Cooley-Turkey 고속 퓨리어 변환 (fast Fourier transform: FFT) 의 버터플라이 연산흐름도와 동일한 규칙적인 연산구조를 갖으며, 본 논문에서는 이를 2차원 배열에 매핑하여 효율적인 VLSI 구현이 가능하도록 하였다.

II. Vector-Radix 2차원 고속 DCT (VR-FCT) 알고리듬

입력 데이타 {x(i,j)}에 대한 (N x N) 2차원 DCT의 계수 {X(u,v)}는 다음과 같이 정의된다.[1]

$$X(u,v) = \frac{4\varepsilon_u\varepsilon_v}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}x(i,j)\cos\frac{\pi(2i+1)u}{2N} \cdot \cos\frac{\pi(2j+1)v}{2N} \quad (1)$$

단, $\varepsilon_u, \varepsilon_v = \begin{cases} 1/\sqrt{2} & \text{for } u,\ v=0 \\ 1 & \text{otherwise.} \end{cases}$

편의상, scale factor는 X(u,v)에 포함되며, $N=2^k$ (단, k는 양의 정수) 이라고 가정한다. 식(1)에 식(2)의 인덱스 매핑을 적용한 후, 인덱스 (u,v)를 짝수-짝수, 짝수-홀수, 홀수-짝수, 홀수-홀수 4개의 영역으로 분할하여 4개의 (N/2 x N/2) DCT로 분할하고, 각각의 분할된 DCT에 대해 (2 x 2) DCT가 얻어질 때까지 동일한 분할과정을 반복하면 식(3)이 얻어진다.[13]

$$f(i,j) = \begin{cases} x(2i,2j) & i=0,\cdots,N/2-1; j=0,\cdots,N/2-1 \\ x(2N-2i-1,2j) & i=N/2,\cdots,N-1; j=0,\cdots,N/2-1 \\ x(2i,2N-2j-1) & i=0,\cdots,N/2-1; j=N/2,\cdots,N-1 \\ x(2N-2i-1,2N-2j-1) & i=N/2,\cdots,N-1; j=N/2,\cdots,N-1 \end{cases} \quad (2)$$

i) 짝수-짝수 인덱스 부분:

$$X(2u,2v) = \sum_{i=0}^{N/2-1}\sum_{j=0}^{N/2-1} f^{00}(i,j) \cdot \cos\frac{\pi(4i+1)u}{2(N/2)} \cdot \cos\frac{\pi(4j+1)v}{2(N/2)} \quad (3\text{-a})$$

ii) 홀수-짝수 인덱스 부분:

$$X(2u+1,2v) = \sum_{i=0}^{N/2-1}\sum_{j=0}^{N/2-1}[\ 2f^{10}(i,j)\cos\varphi_1] \cdot \cos\frac{\pi(4i+1)u}{2(N/2)}\cos\frac{\pi(4j+1)v}{2(N/2)} - X(2u-1,2v) \quad (3\text{-b})$$

iii) 짝수-홀수 인덱스 부분:

$$X(2u,2v+1) = \sum_{i=0}^{N/2-1}\sum_{j=0}^{N/2-1}[\ 2f^{01}(i,j)\cos\varphi_2] \cdot \cos\frac{\pi(4i+1)u}{2(N/2)}\cos\frac{\pi(4j+1)v}{2(N/2)} - X(2u,2v-1) \quad (3\text{-c})$$

iV) 홀수-홀수 인덱스 부분:

$$X(2u+1,2v+1) = \sum_{i=0}^{N/2-1}\sum_{j=0}^{N/2-1}[\ 4f^{11}(i,j)\cos\varphi_1\cos\varphi_2]\ \cos\frac{\pi(4i+1)u}{2(N/2)} \cdot \cos\frac{\pi(4j+1)v}{2(N/2)} - X(2u-1,2v+1) - X(2u+1,2v-1) - X(2u-1,2v-1) \quad (3\text{-d})$$

단, $f^{ab}(i,j) = f(i,j) + (-1)^a \cdot f(i+\frac{N}{2},j) + (-1)^b \cdot f(i,j+\frac{N}{2}) + (-1)^{a+b} \cdot f(i+\frac{N}{2},j+\frac{N}{2})$ for $a,\ b=0$ or 1

$$\cos\varphi_1 = \cos\frac{\pi(4i+1)}{2N},\ \cos\varphi_2 = \cos\frac{\pi(4j+1)}{2N}$$
$$\text{for } i,\ j=0,\ 1,\cdots,\ \frac{N}{2}-1.$$

식(3)에 의하면, (N x N) 2차원 VR-FCT는 $\log_2 N$ stages의 버터플라이 연산과정과 후처리 가산과정에 의해 계산되며, 버터플라이 연산부분은 (N^2)-point 1차원 FFT와 동일한 연산구조를 갖는다.

Ⅲ. VR-FCT의 VLSI 병렬계산을 위한 어레이 알고리듬

II장에서 VR-FCT 알고리듬이 규칙적인 버터플라이 연산과정과 후처리 가산과정으로 구성됨을 설명하였다. 본 장에서는 식(3)에 포함된 버터플라이 연산과정을 2차원 어레이에 매핑하여 병렬계산하기 위한 효율적인 어레이 알고리듬을 제안하고자 한다.

식(3)에서 버터플라이 연산부분을 짝수-짝수, 홀수-짝수, 짝수-홀수, 홀수-홀수 등 4개의 부분 인덱스 영역으로 나누어 표현하면 식(4)와 같이 된다.

i) 짝수-짝수 인덱스 부분;

$$\begin{aligned} f_q(i_1,j_1) = \ & C_p\cdot[\{f_{q-1}(i_{1,j_1})+f_{q-1}(i_1+j_1)\} \\ & +\{f_{q-1}(i_1,j_1+D_q) \\ & +f_{q-1}(i_1+D_q,j_1+D_q)\}] \\ & \text{단}, C_p=1 \end{aligned} \tag{4-a}$$

ii) 홀수-짝수 인덱스 부분;

$$\begin{aligned} f_q(i_1+D_q,j_1) = & 2C_p[\{f_{q-1}(i_1,j_1)-f_{q-1}(i_1+D_q,j_1)\} \\ & +\{f_{q-1}(i_1,j_1+D_q)-f_{q-1}(i_1+D_q,j_1+D_q)\}] \\ & \text{단}, C_p=\cos\{\pi(4i_1+1)(2^{q-2}/N)\} \end{aligned} \tag{4-b}$$

iii) 짝수-홀수 인덱스 부분;

$$\begin{aligned} f_q(i_1,j_1+D_q) = 2C_p[& \{f_{q-1}(i_1,j_1)\} \\ & \{+f_{q-1}(i_1+D_q,j_1)\} \\ & -\{f_{q-1}(i_1,j_1+D_q)\} \\ & \{+f_{q-1}(i_1+D_q,j_1+D_q)\}] \\ \text{단}, C_p=\cos\{\pi(4j_1+1)(2^{q-2}/N)\} & \end{aligned} \tag{4-c}$$

iv) 홀수-홀수 인덱스 부분;

$$\begin{aligned} f_q(i_1+D_q,j_1+D_q) = 4C_p[& \{f_{q-1}(i_1,j_1) \\ & -f_{q-1}(i_1+D_q,j_1)\} \\ & -\{f_{q-1}(i_1,j_1+D_q) \\ & -f_{q-1}(i_1+D_q,j_1+D_q)\}] \\ \text{단}, C_p=\cos\{\pi(4i_1+1)(2^{q-2}/N)\} & \\ \cdot\cos\{\pi(4j_1+1)(2^{q-2}/N)\} & \end{aligned} \tag{4-d}$$

식(4)에서 Dq는 q-번째 버터플라이 stage에서의 데이타 셔플링 거리를 나타내며, 식(5)와 같이 주어진다. 또한, 인덱스 i_1, j_1은 각각 식(6)을 만족하는 인덱스 i, j의 조합으로 이루어진다.

$$D_q = 2^{\log_2 N - q} \tag{5}$$

$$i \ \ MOD \ \ 2^{\log_2 N-q+1} < 2^{\log_2 N-q} \tag{6-a}$$

$$j \ \ MOD \ \ 2^{\log_2 N-q+1} < 2^{\log_2 N-q} \tag{6-b}$$

식(4)를 고찰하여 보면, 중괄호 안의 연산 {·}은 행 인덱스가 D_q 만큼 떨어진 데이타 짝에 대해 이루어지며, 대괄호 안의 연산 [·]은 열 인덱스가 D_q 만큼 떨어진 데이타 짝에 대해 이루어짐을 알 수 있다. 따라서, 데이타 f(i,j)를 2차원 어레이에 매핑하면 중괄호 안의 연산 {·}은 열방향의 데이타 이동을 수반하고, 대괄호 안의 연산 [·]은 행방향의 데이타 이동을 수반한다. 이와 같은 사실로 부터, 식(4)를 2차원 어레이에 매핑하여 처리하면 VR-FCT의 효율적인 병렬연산이 가능함을 알 수 있으며, 본 논문에서는 이와 같은 사실을 근거로하여 VR-FCT의 병렬처리 알고리듬을 제안하고자 한다. (N x N) VR-FCT의 병렬계산을 위한 2차원 어레이는 N행 x N열의 처리요소들로 구성되며, 데이타 f(i,j)는 처리요소의 PE(i,j)에 1 : 1 매핑된다. 데이타가 어레이에 입력된 후, 데이타 셔플링과 버터플라이 연산이 $\log_2 N$ stages에 걸쳐 계산된 후, 후처리 가산과정을 거쳐 최종결과 {X(u,v)}가 출력된다. 각 버터플라이 연산 stage는 열방향 처리와 행방향 처리의 두단계 과정으로 이루어진다. 열방향 처리에서는 식(5)에 주어진 셔플링 거리 D_q 만큼 데이타가 열방향으로 이동된 후, 식(4)의 중괄호 안의 연산 {·} (즉, 버터플라이 데이타 짝에 대한 '+' 또는 '-'연산)이 이루어진다. 한편, 행방향 처리에서는 셔플링 거리 D_q 만큼 데이타가 행방향으로 이동된 후, 식(4)의 대괄호 안의 연산 [·] (즉, 버터플라이 데이타 짝에 대한

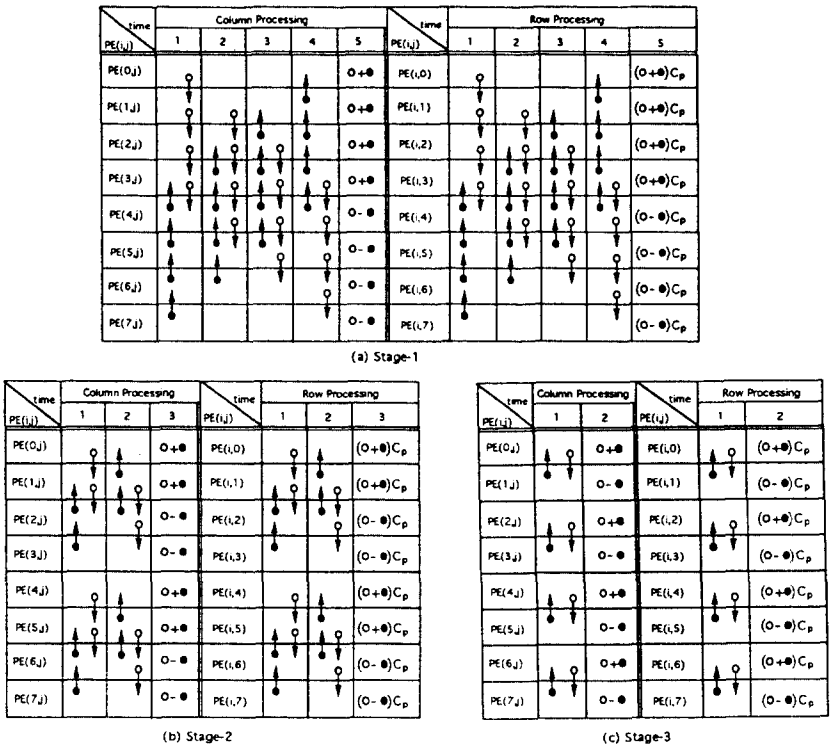

그림 1. (8 x 8) 2차원 VR-FCT에 대한 ALGORITHM-I의 연산과정
Fig. 1. Illustrations of the ALGORITHM-I for (8 x 8) 2-D VR-FCT.

```
BEGIN          /* for all i and j (0 ≤ i,j ≤ N-1) */
  k=log2N
  FOR q=1 TO log2N DO IN PARALLEL
      Dq = 2^(k-q)          /* Shuffling Distance */
      i1 = i MOD 2^(k+1-q)  /* Column Processing */
      IF (i1 < 2^(k-q)) THEN
         move data fq-1(i1,j) to index point (i1+Dq,j)
         PE(i1,j) compute fq(i1,j)=fq-1(i1,j)+fq-1(i1+Dq,j)
      ELSE
         move data fq-1(i1,j) to index point (i1-Dq,j)
         PE(i1,j) compute fq(i1,j)=fq-1(i1-Dq,j)-fq-1(i1,j)
      ENDIF
      j1 = j MOD 2^(k+1-q)  /* Row Processing */
      IF (j1 < 2^(k-q)) THEN
         move data fq(i,j1) to index point (i,j1+Dq)
         PE(i,j1) compute fq(i,j1)=[fq(i,j1)+fq(i,j1+Dq)]Cp
      ELSE
         move data fq(i,j1) to index point (i,j1-Dq)
         PE(i,j1) compute fq(i,j1)=[fq(i,j1-Dq)-fq(i,j1)]Cp
      ENDIF
  ENDFOR
  post-processing stage
END.
```

알고리듬-I. 2차원 배열을 이용한 (N×N) 2-D VR-FCT계산 알고리듬
ALGORITHM-I. Proposed array algorithm for computing (N×N) 2-D VR-FCT using 2-D array.

'+' 또는 '-'연산과 C_p값 곱셈연산)이 이루어진다. 2차원 어레이상에서 식(4)가 계산되는 과정을 알고리듬으로 표현한 것이 ALGORITHM-I 이며, 그림 1은 (8 x 8) VR-FCT가 ALGORITHM-I에 의해 계산되는 과정을 보인 것이다. 버터플라이 연산과정은 $\log_2 8=3$ stages로 구성되며, 각 연산 stage는 열방향 처리와 행방향 처리의 두단계 과정으로 이루어진다. stage-1의 연산과정은 다음과 같다. 열방향 처리과정에서는 데이타가 열방향으로 셔플링거리 $D_1=2^{3-1}=4$ 만큼 떨어진 처리요소로 이동된 후, 버터플라이 데이타 짝 (각 처리요소에 초기입력된 데이타와 이동된 데이타)에 대한 '+'연산 또는 '-'연산이 이루어 진다. 행방향 처리과정에서는 열방향 처리에 의해 생성된 데이타가 행방향으로 셔플링거리 $D_1=2^{3-1}=4$ 만큼 떨어진 처리요소로 이동된 후, 버터플라이 데이타 짝 (즉, 열방향 처리에 의해 생성된 각 처리요소내의 데이타와 이동된 데이타)에 대한 '+'연산 또는 '-'연산과 C_p값 곱셈이 이루어진다. stage-1의 데이타 셔플링은 셔플링 거리가 $D_1=4$ 이므로 4 클럭이 소요되며, 어레이를 구성하는 처리요소중 절반은 '+'연산을 수행하고, 나

머지 절반은 '-'연산을 수행한다. stage-2와 3의 연산도 동일한 방법으로 이루어지며, stage-2의 셔플링거리는 $D_2=2$ 이고, stage-3의 셔플링거리는 $D_3=1$이 된다. 한편, 식(4)에서 C_p값은 버터플라이 stage q와 처리요소의 인덱스 (i,j)에 의해 결정된다.

본 논문에서 제안된 알고리듬으로 (N x N)-point 2차원 DCT를 계산하는데 소요되는 시간 (T_{total})은 $\log_2 N$ stage에 걸친 데이타 셔플링 시간 ($T_{shuffling}$)과 버터플라이 연산시간 ($T_{butterfly}$)의 합이 된다. $\log_2 N$ stages에 걸친 데이타 셔플링은 행방향 처리와 열방향 처리에서 모두 이루어지므로, 총 셔플링 시간은 식(5)가 나타내는 셔플링거리를 $\log_2 N$ stages 만큼 합한 것의 두배가 되며, 식 (7-a)와 같이 주어진다. 한편, 버터플라이 연산에서 '+'연산 또는 '-'연산은 행방향 처리와 열방향 처리에서 모두 이루어지며 C_p값 곱셈은 행방향 처리에서만 이루어지므로, 총 버터플라이 연산시간은 식 (7-b)와 같이 주어진다. 식(7-a)와 식(7-b)로 부터, (N x N) 2차원 DCT의 연산시간은 식(8)과 같이 표현된다. 식(8)로 부터, (N x N) 2차원 DCT에 대해 $O(N+N_{NZD}\cdot\log_2 N)$의 시간복잡도를 갖음을 알 수 있다. 식(7),(8)에서 N_{NZD}는 C_P값을 CSD 코드로 표현했을 때의 non-zero digit의 수를 나타내며, T는 클럭주기이다.

$$T_{shuffling} = 2T\sum_{q=1}^{\log_2 N}(D_q+1) = 2T\cdot(N+\log_2 N-1) \quad (7\text{-a})$$

$$T_{butterfly} = (4+N_{NZD})\log_2 N\cdot T \quad (7\text{-b})$$

$$T_{total} = [\ 2(N-1)+(6+N_{NZD})\log_2 N]\ \cdot T \quad (8)$$

IV. 회로설계

III장에서 제안된 ALGORITHM-I은 2차원 배열을 기본으로 하며, 배열을 구성하는 처리요소들은 동일한 기능을 수행한다. 따라서, 전체적인 회로설계는 기본 처리요소 하나의 설계로 단순화된다. 본 장에서는 어레이를 구성하는 기본 처리요소의 회로설계 및 레이아웃 설계에 대해 언급하고자 한다.

1. 기본 처리요소 설계

2차원 어레이를 구성하는 기본 처리요소의 내부구성도는 그림 2와 같으며, 데이타 셔플링 회로, 버터플라이 연산회로, C_p값 저장 및 제어신호 발생회로 등의 기능블럭으로 구성된다.

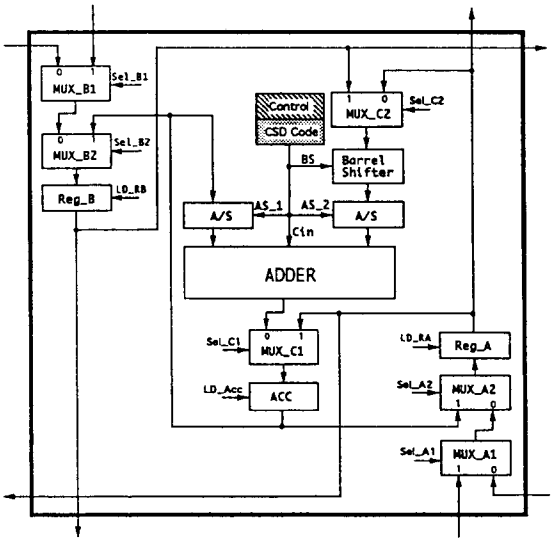

그림 2. 처리요소의 내부구성도
Fig. 2. Internal block diagram of processing element.

데이타 셔플링회로는 인접한 처리요소와의 데이타 교환기능을 수행하며, 멀티플렉서와 파이프라인 레지스터로 구성된다. 버터플라이 연산회로는 식(4)가 나타내는 연산 즉, $(A+B)\cdot C_p$ 또는 $(A-B)\cdot C_p$를 계산하는 연산회로이며, 가산기만을 사용하여 쉬프트-가산방식으로 구현하였다. 특히, 쉬프트-가산방식의 연산속도를 개선하기 위하여 canonic-signed digit (CSD)[18] 연산을 적용하였다. 즉, 처리요소의 위치와 버터플라이 stage q에 따라 결정되는 C_p값은 2의 보수표현 대신에 (-1,0,1)로 표현되는 ternary 코드로 변환되어 처리요소내에 저장되고, 쉬프트-가산연산의 쉬프트제어에 사용된다. 이와 같이 CSD 연산을 사용함으로써 2의 보수연산을 사용하는 경우 보다 약 30% 정도의 가산횟수를 줄일 수 있었다. CSD 코드를 이용한 쉬프트-가산연산을 구현하기 위해서는 CSD 코드내의 non-zero digit의 위치에 따라 데이타를 좌측 또는 우측으로 쉬프트시키는 기능이 필요하다. 본 논문에서는 한번에 여러비트를 쉬프트시킬 수 있는 배럴쉬프터를 사용하였으며, 설계된 배럴쉬프터는 우측으로 13-비트, 좌측으로 2-비트 쉬프트기능을 갖는다. 또한, 우측으로 쉬프트할때에는 부호확장 (sign extension) 기능을 갖으며, 좌측으로 쉬프트할때에는 최하위 비트에 영삽입 (zero insertion) 기능을 갖도록 설계하였다.

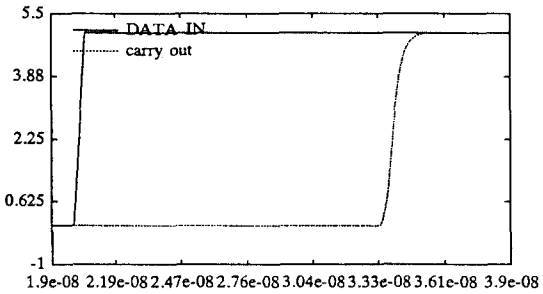

그림 3. 처리요소내의 최대지연경로에 대한 SPICE 시뮬레이션 결과(ISRC 1.5μm N-Well CMOS 파라메터 사용)
Fig. 3. SPICE simulation result for critical delay path of PE. (using ISRC 1.5μm N-Well CMOS parameters)

처리요소를 구성하는 회로요소중 전체 칩의 성능(즉, 면적과 동작속도)에 가장 큰 영향을 미치는 부분은 버터플라이 연산블럭내의 가산기 회로이다. 전체 어레이를 단일칩에 집적시키기 위해서는 기본 처리요소의 면적을 최소화하는 것이 중요한 요소가 되며, 본 논문에서는 가산기 회로의 선택기준으로 동작속도 보다는 면적에 비중을 두어 ripple-carry 가산기를 채택하였다. 처리요소내의 최대지연경로는 데이타 레지스터(Reg_A) 에서부터 배럴쉬프터와 가산기를 거쳐 누산기 (Acc) 에 이르는 경로이다.(그림 2 참조) 반도체공동연구소(ISRC)의 1.5μm N-well CMOS 공정 파라메터를 사용하여 SPICE 시뮬레이션 결과는 그림 3과 같으며, 최대지연시간은 약 16 nsec로 나타났다. 이와 같은 시뮬레이션 결과를 토대로, 처리요소의 동작클럭 주파수를 50 MHz로 결정하였다. 처리요소의 동작을 제어하기 위한 제어신호의 타이밍도는 그림 4와 같으며, 데이타셔플링을 제어하는 기능과 버터플라이 연산을 제어하는 기능을 갖는다. 설계된 처리요소의 논리기능을 논리 시뮬레이터 SILOS[19]를 이용하여 검증하였다.

2. 레이아웃 설계

설계된 회로를 집적회로로 구현하기 위한 레이아웃 설계는 반도체공동연구소 (ISRC) 1.5μm N-Well CMOS 설계규칙을 이용하여 수행하였다. 레이아웃 설계는 설계복잡도를 줄이고 설계시간을 단축하기 위하여 Bit Slice Unit (BSU)를 기본으로한 계층적 방식을 이용하였다. 그림 2의 처리요소 회로에서 배럴쉬프터를 제외한 나머지 부분을 비트단위로 분할하여 단위 BSU를 정의하고, 정의된 BSU를 다시 5개의 매크로셀로 분할하여 설계하였다. 단위 BSU의 floor plan은 그림 5-(a)와 같이 5개의 매크로셀을 수직으로 배치한 형태가 되도록 하였다. 특히, BSU 설계시에는 레이아웃 계층사이의 배선 및 연결이 단순화되도록 BSU의 외부 배선단자의 위치를 규정화 하였다. 즉, 인접한 BSU 사이의 연결 및 인접한 처리요소 사이의 연결이 추가적인 배선없이 단순히 BSU 및 처리요소의 반복구조에 의해 완성될 수 있도록 하였다. 설계된 BSU의 레이아웃은 그림 5-(b)와 같으며, 면적은 159μm x 717μm 이다. 한편, 비트단위로 분할이 불가능한 배럴쉬프터 부분은 단일 매크로블럭으로 설계하였다. 처리요소의 레이아웃은 내부 데이타 비트수 만큼 BSU를 반복시키고 배럴쉬프터를 삽입함으로써 처리요소의 레이아웃을

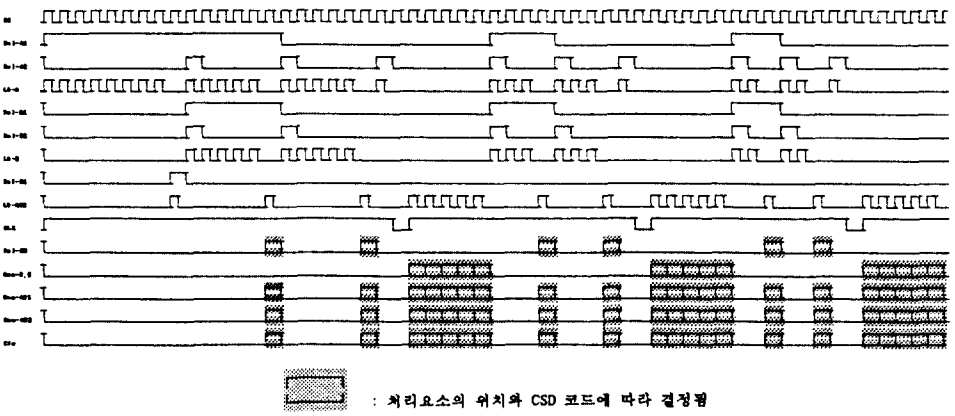

그림 4. 제어신호의 타이밍도
Fig. 4. Tiing diagram for control signals.

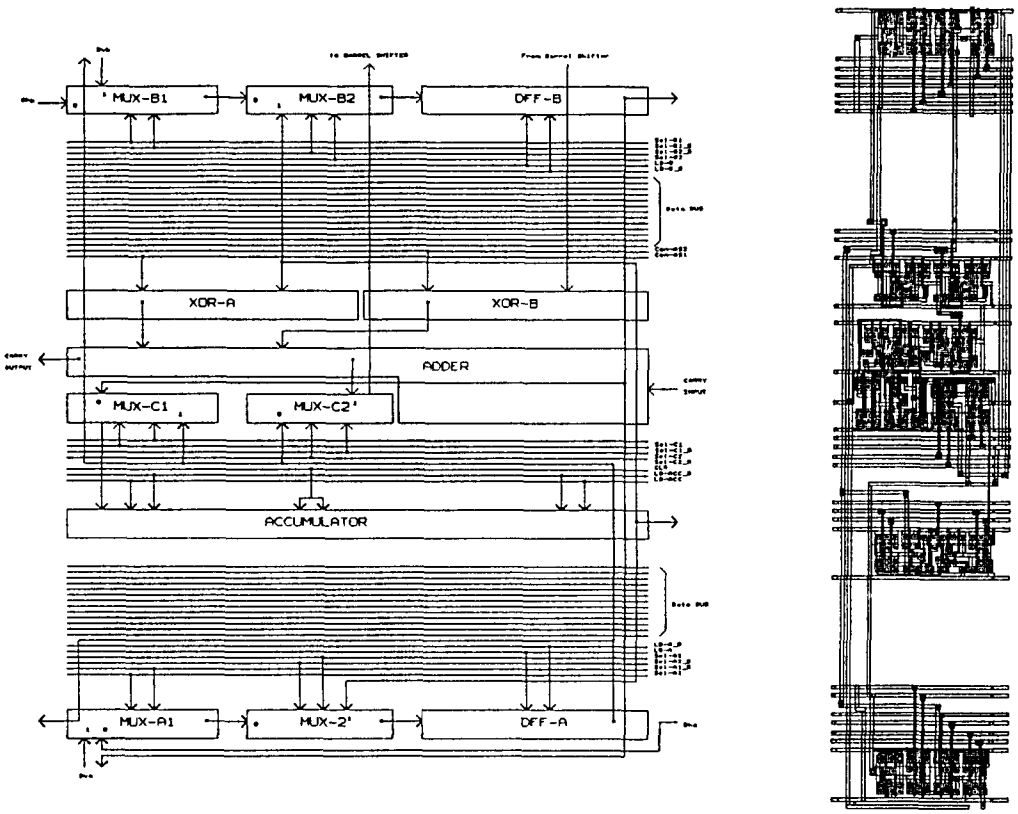

그림 5. BSU에 대한 레이아웃
Fig. 5. Floor plan and layout of Bit-Slice Unit (BSU).

완성하였다. 완성된 처리요소의 레이아웃은 그림 6과 같으며, 면적은 1985μm x 1069μm 이다.

표 1. 레이아웃 결과 요약
Table 1. Layout statistics.

Block	Area(㎟)	Transistors
Bit-Slice Unit	0.110	102
13-bit Barrel Shifter	0.617	206
Processing Element	2.122	1,532
8 X 8 Array Core*	138.110	98,000
Technology	ISRC 1.5μm double metal N-Well CMOS	

*Excluding control logic block

한편, 어레이 코어의 레이아웃은 처리요소 64개를 8 x 8로 배열함으로써 완성되며, 그림 7은 제어회로 부분이 포함되지 않은 (8 x 8) 2차원 어레이 코어의 레이아웃이다. 완성된 레이아웃 데이타로부터 회로연결정보를 추출하고 이를 이용하여 스위치레벨 논리시뮬레이션을 수행함으로써 레이아웃 설계를 검증하였다. 표(1)은 레이아웃 설계결과를 요약한 것이다.

V. 결 론

본 논문에서는 vector-radix 2차원 고속 DCT (VR-FCT)를 VLSI 병렬계산하기 위한 효율적인 어레이 알고리듬을 제안하고, 이를 집적회로로 구현하기 위한 회로를 설계하였다. 제안된 DCT 구현방법의 특징은 다음과 같다. 첫째, VR-FCT 알고리듬의 버터플라이 연산부분을 2차원 어레이에 매핑하여 이를 병렬 및 파이프라인 처리함으로써 VR-FCT 알고리듬의 고속성과 2차원 어레이의 병렬성 및 국부통신 특성을 동시에 이용할 수 있다.

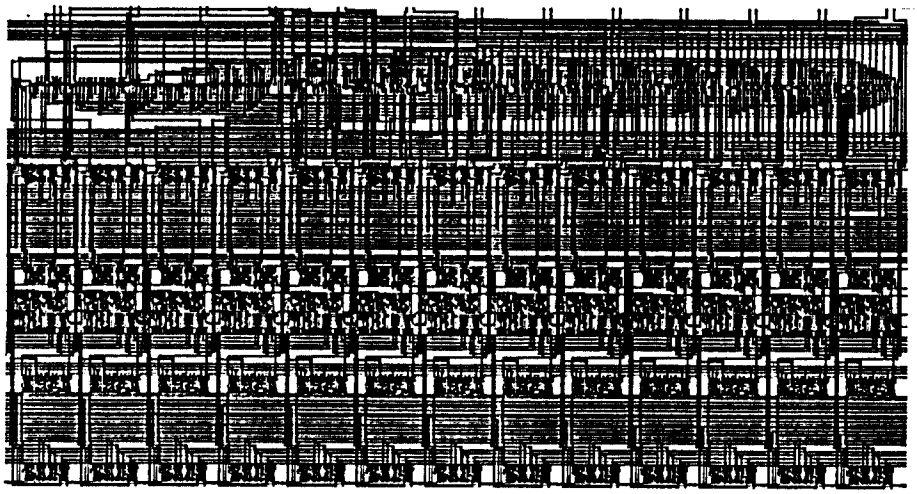

그림 6. 처리요소의 레이아웃
Fig. 6. Layout of processing element.

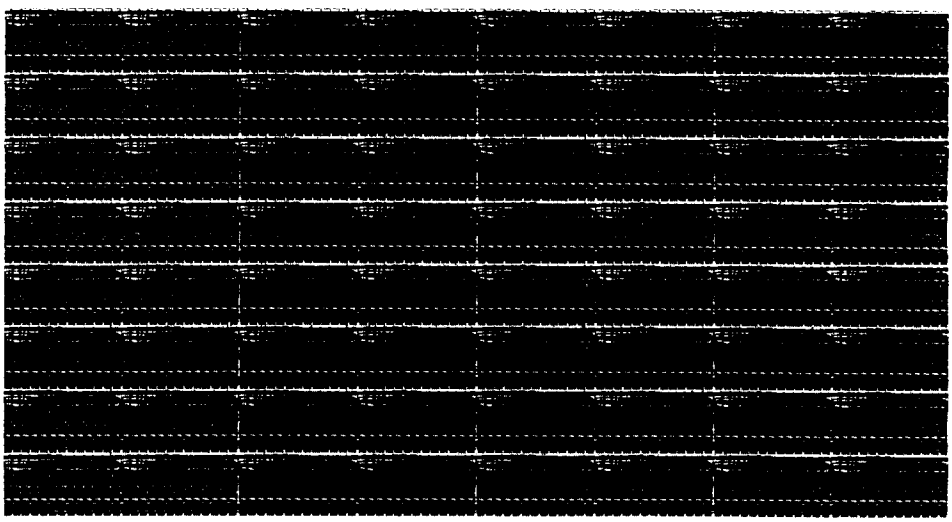

그림 7. (8 x 8) 2차원 어레이 코어의 레이아웃 (제어회로 제외)
Fig. 7. Layout of (8 x 8) 2-D array core (Excluding control logic block).

둘째, matrix transposition 메모리가 필요치 않다.

셋째, 2차원 어레이의 구조적 특징인 규칙성, 모듈성 및 국부연결성 등에 의해 VLSI 구현에 매우 적합하다.

넷째, (NxN) 2차원 DCT에 대해 $O(N+N_{NZP}\log_2 N)$의 시간복잡도를 갖는다. 제안된 어레이 알고리듬을 집적회로로 구현하기 위해 ISRC 1.5μm 이중금속배선 N-Well CMOS 공정으로 어레이 코어를 설계하였다.

설계된 어레이 코어는 64개의 처리요소가 (8 x 8) 2차원 배열로 구성되며, 약 $138mm^2$의 면적에 98,000여개의 트랜지스터로 구성된다. 50 MHz 클럭주파수에서 (8 x 8) DCT 계산에 약 0.88 μsec가 소요되며, 약 $72\cdot10^6 pixels/\sec$의 연산성능이 예상된다. 따라서, 실시간 영상처리를 위한 2차원 DCT 프로세서의 기본 아키텍처로 이용가능할 것으로 평가된다.

참 고 문 헌

[1] K.R. Rao and P. Yip, Discrete Cosine Transform: Algorithms, Advantages, Applica tions, Academic Press, Inc., 19 90.

[2] G.K. Wallace, "The JPEG still-picture compression standard", Communications of the ACM, vol.34, no.4, pp.31-44, Apr., 1991.

[3] D.Le Gall, "MPEG : A video compression standard for multimedia applications", Communications of the ACM, vol.34, no.4, pp.47-58, Apr., 1991.

[4] K.B. Benson and D.G. Flink, HDTV Advanced television for the 1990s, McGraw Hill, 1991.

[5] S. Wolter, D. Birreck and R. Laur, "Classification for 2D DCT's and a new archtecture with distributed arithmetic", 1991 IEEE International Symposium on Circuits and Systems (ISCAS '91), pp.2204-2207, 1991.

[6] N.I. Cho and S.U. Lee, "DCT algorithms for VLSI parallel implementations", IEEE Trans. on Acoustics, Speech, and Signal Processing, vol. ASSP-38, no.1, pp.121-127, Jan., 1990.

[7] J. Carlach, P. Penard and J. Sicre, "TCAD : a 27MHz 8 x 8 discrete cosine transform chip", 1989 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '89), pp.2429-2432, 1989

[8] A. Artieri et al, "A one chip VLSI for real time two-dimensional discrete cosine transform", 1988 IEEE International Symposium on Circuits and Systems (IS CAS '88), pp.701-704, 1988.

[9] B. Sikstroem, et al. "A high speed 2-D discrete cosine transform chip", *Integration VLSI Journal* 5, pp.159-169, 1987.

[10] M.T. Sun, T.C. Chen and A.M. Gottlieb, "VLSI implementation of a 16 x 16 discrete cosine transform",IEEE Trans. on Circuits and Systems, vol.CAS-36, no.4, pp.610-617, Apr., 1989.

[11] U. Sjoestroem et al, "Discrete cosine transform chip for real-time video applications", 1990 IEEE International Symposium on Circuits and Systems (ISCAS '88), pp.1620-1623, 1990.

[12] M.A. Haque, "A two-dimensional fast cosine transform", IEEE Trans. on Acoustics, Speech, and Signal Processing, vol.33, no.6, pp.1532-1536, Dec, 1985.

[13] S.C. Chan and K.L. Ho, "A new two-dimensional fast cosine transform algorithm", IEEE Trans. on Signal Processing, vol.39, no.2, pp.481-485, Feb., 1991.

[14] P. Duhamel, C. Guillemot, "Polynomial transform computation of the 2-D DCT", 1990 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '90), pp.15 15-1518, 1990.

[15] N.I. Cho and S.U. Lee, "Fast algorithm and implementation of 2-D discrete cosine transform", IEEE Trans. on Circuits and Systems, vol.38, no.3, pp.297-305,Mar., 1991.

[16] M. Yan and J.V. McCanny, "VLSI architecture for computing the 2-D DCT", International Conference on Systolic Arrays, pp.411-429, 1989.

[17] M. K. Lee, K. W. Shin and J.K Lee, "A VLSI array processor for 16-point FFT", *IEEE J. of Solid-State Circuits*, vol. SC-26, no. 9, pp.1286-1292, Sep., 1991.

[18] K. Hwang, Computer arithmetic: Principles, Architecture and Design, John Wiley & Sons, 1979.

[19] P/C SILOS: Logic Simulator, SIMUCAD, Inc., 1988.

[20] K.W. Shin, H.W. Jeon, Y.S. Kang, "An efficient VLSI implementation of vectorradix 2-D DCT using mesh-connected

2-D array". 1994 IEEE International Symposium on Circuits and Systems (ISCAS'94), vol.4, pp.47-50, 1994.

저 자 소 개

姜 龍 暹(正會員)

1965년 1월 11일생. 1992년 2월 금오공과대학교 전자공학과 졸업 (공학사). 1994년 2월 금오공과대학교 대학원 전자공학과 졸업 (공학석사). 1994년 3월 - 현재 고려대학교 전자공학과 한국 과학기술원 (KIST) 응용전자부 학·연 박사과정 재학중. 주관심 분야는 ASIC 설계, VLSI 설계, VLSI Architectures.

全 興 雨(正會員)

1956년 10월 30일생. 1980년 2월 한국항공대학 전자공학과 졸업 (공학사). 1982년 2월 고려대학교 대학원 전자공학과 졸업 (공학석사). 1988년 8월 고려대학교 대학원 전자공학과 졸업 (공학박사). 1989년 3월 - 현재 금오공과대학교 전자공학과 조교수. 주관심 분야는 VLSI 설계 및 CAD

辛 卿 旭(正會員)

1961년 10월 26일생. 1984년 2월 한국항공대학 전자공학과 졸업 (공학사). 1986년 2월 연세대학교 대학원 전자공학과 졸업 (공학석사). 1990년 8월 연세대학교 대학원 전자공학과 졸업 (공학박사). 1990년 9월 - 1991년 6월 한국전자통신연구소 반도체연구단 선임연구원. 1991년 7월 - 현재 금오공과대학교 전자공학과 조교수. 주관심 분야는 통신 및 신호처리용 디지탈/아날로그 집적회로 설계, 저전압/저전력 집적회로 설계