재구성 가능한 DCT/DWT 프로세서 설계

김영진° 이현수
경희대학교 전자정보대학
jerryjin@nate.com, leehs@khu.ac.kr

The Reconfigurable Processor Design of DCT/DWT

YoungJin Kim° Hyon Soo Lee
College of Electronics and Information, Kyung-hee University.

요 약

최근 이미지 압축, 워터마킹 또는 스케일러블 비디오 코딩 분야에서 DCT와 DWT 연산을 선택적으로 사용하거나, 혼합하여 사용하는 경우가 늘어나고 있다. 이러한 두개의 연산을 사용하는 방법은 소프트웨어적인 프로그램을 사용하거나 하드웨어를 따로 구현하여 사용하였다. 본 연구에서는 하나의 모듈로 두개의 연산을 수행할 수 있는 재구성 하드웨어를 제안한다. 또한 DCT와 DWT연산에 있어서, 가장 많은 연산을 수행하는 부분은 계수(Coefficient)값과 입력 값의 내적 연산(Inner Product)을 수행하는 것인데, 이 내적연산을 하는데 있어서 곱셈기를 사용하지 않는 분산연산을 사용함으로써 연산의 복잡도를 줄이고, 하드웨어의 속도를 빠르게 하였다. 실험 환경은 Altera FPGA를 사용한 Excalibur_ARM (EPXA10F1020C1) 보드를 이용하여 구현하였으며, 동작속도는 47.85MHz이다.

1. 서 론

DCT(Discrete Cosine Transform)와 DWT(Discrete Wavelet Transform)연산은 영상처리, 영상 압축, 신호분석에 널리 사용되는 알고리즘이다. DCT 연산은 시간적, 공간적 중복성을 제거 하고, 압축효율을 높이는 장점이 있지만, 압축율이 높을 경우 블록킹 현상이 발생하여 이미지의 품질을 떨어뜨리는 단점이 있다. 반면, DWT 연산은 이미지 중에서 중요한 정보가 부대역(Subband)영역으로 집중하는 효과가 있어 블록킹 현상을 해결할 수 있으며, 다해상도(Multiresolution Analysis: MRA)로 분할되는 특징을 가진다. 또한 인간의 시각 특성(HVS: Human Visual System)에 가깝다는 장점이 있다.

최근에는 이미지 압축(Image Compression)[1], 워터마킹(Watermarking)[2] 또는 스케일러블 비디오 코딩(Scalable Video Coding)[3] 등의 분야에서 DCT와 DWT의 장점을 사용하기 위해 두 개의 연산을 선택적으로 사용하거나, 같이 사용하는 경우가 있다. 이러한 분야에서 사용되는 하드웨어는 DCT와 DWT를 각각 따로 구현한 것을 사용한다[1][3]. 하지만, 각각의 모듈을 따로 구현하였을 때, 하드웨어의 크기가 커지며, 데이터의 교환으로 인해서 연산시간이 길어진다는 단점을 가지고 있다. 하드웨어의 성능 향상을 위한 방법으로는 시스톨릭 구조[4][5]나 병렬 구조[6]를 사용하는 것이다. 이 구조들은 DCT와 DWT의 내적 연산에서 가장 큰 복잡도를 갖는 곱셈기의 수를 줄이는 데는 한계가 있다. 이를 개선하기 위해서 분산연산(DA: Distributed Arithmetic)을 사용한다[7]. 분산연산은 내적 연산을 수행하는데 곱셈기를 사용하지 않고, bit별 연산에 대한 결과를 LUT(Look-Up Table)형식으로 계산하는 방법이다.

본 논문에서는 하나의 프로세서를 연산에 따라 재구성함으로써 DCT연산과 3단계 DWT연산을 모두 수행하는 프로세서 구조를 제안한다. 또한 내적에 대한 곱셈기의 연산 복잡도를 줄이기 위해 롬-기반 분산연산(ROM-based Distributed Arithmetic: ROM-based DA)을 사용하였다.

2. DCT/ DWT 알고리즘

DCT와 DWT 연산은 계수(Coefficient)와 입력값의 내적 연산을 수행한다는 공통적 특징을 가지고 있다.

2.1 DCT 알고리즘

1차원 DCT 연산은 식 (1)과 같다.

$$F(u)=\frac{2}{N}C(u)\sum_{x=0}^{N-1}f(x)\cos\frac{(2x+1)u\pi}{2N} \quad (1)$$

$$where\begin{cases} C(u)=\frac{1}{\sqrt{2}} & \text{if } u=0 \\ C(u)=1 & otherwise \end{cases}$$

여기서 $f(x)$는 입력이고 $F(u)$는 DCT연산의 결과이다. 연산의 수를 줄이기 위해서 Chen의 알고리즘을 사용하면 식(2), (3)과 같이 줄일수 있으며, 아래의 식은 8-포인트 1차원(8-Point 1-Dimension)연산에 대한 식이다.

$$\begin{bmatrix}X_0\\X_2\\X_4\\X_6\end{bmatrix}=\begin{bmatrix}C_4 & C_4 & C_4 & C_4\\ C_2 & C_6 & -C_6 & -C_2\\ C_4 & -C_4 & -C_4 & C_4\\ C_6 & -C_2 & C_2 & -C_6\end{bmatrix}\begin{bmatrix}x_0+x_7\\x_1+x_6\\x_2+x_5\\x_3+x_4\end{bmatrix} \quad (2)$$

$$\begin{bmatrix}X_1\\X_3\\X_5\\X_7\end{bmatrix}=\begin{bmatrix}C_1 & C_3 & C_5 & C_7\\ C_3 & -C_7 & -C_1 & -C_5\\ C_5 & -C_1 & C_7 & C_3\\ C_7 & -C_5 & C_3 & -C_1\end{bmatrix}\begin{bmatrix}x_0-x_7\\x_1-x_6\\x_2-x_5\\x_3-x_4\end{bmatrix} \quad (3)$$

이 식에서 계수에 해당하는 $C_k=\frac{1}{2}\cos\frac{k\pi}{16}$ 이다.

2.2 DWT 알고리즘

DWT 연산을 2개의 부대역(Subband)으로 분해한 일반적인 DWT 3단계 구조는 그림 1과 같고, 각 필터는 FIR 필터를 이용하여 구성할수 있다.

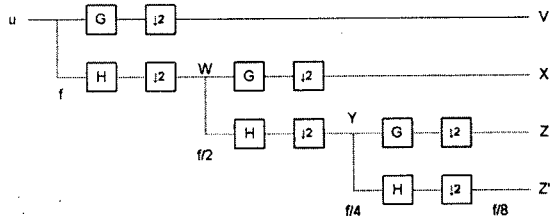

그림 1. 일반적인 DWT 3단계 분해

DWT를 위한 FIR 4차 고대역 통과 필터(Highpass Filter: HPF)와 저대역 통과 필터(Low Pass Filter: LPF)의 변환함수(Transfer Function) G(z), H(z)는 각각 식 (4), (5)와 같다.

$$G(z)=g_0+g_1z^{-1}+g_2z^{-2}+g_3z^{-3} \quad (4)$$

$$H(z)=h_0+h_1z^{-1}+h_2z^{-2}+h_3z^{-3} \quad (5)$$

(z : 지연시간 g_i, h_i : 필터의 계수 값)

3. ROM-based DA 알고리즘

DCT와 DWT의 연산에서 사용되는 내적 연산은 식 (6)과 같다.

$$z=c^tx=\sum_{i=0}^{N-1}c_ix_i, \quad (6)$$

식 (6)에서 i에 해당하는 N은 계수(c)와 입력(x)의 갯수를 의미한다. 내적연산을 bit 별로 연산하기 위해서 입력값을 각 bit에 대한 2의 보수 표현으로 나타내면 식 (7)과 같이 나타낼 수 있다.

$$x_i=-x_{i0}+\sum_{j=1}^{B-1}x_{ij}2^{-j}\ ,\ 0\le i\le N-1 \quad (7)$$

식 (7)에서 j는 bit의 위치, B는 x의 bit 수이다. 분산연산(DA)을 위해서 식 (7)을 식 (6)에 대입하면 식 (8)이 된다.

$$z=-\sum_{i=0}^{N-1}c_ix_{i0}+\sum_{j=1}^{B-1}\left[\sum_{i=0}^{N-1}c_ix_{ij}\right]2^{-j} \quad (8)$$

각 비트 $j=0,1,\ldots,B-1$의 입력에 대해서 $\sum_{i=0}^{N-1}c_ix_{ij}$ 의 값을 롬에 저장하게 된다. 롬의 크기는 연산된 결과를 저장해야하므로, $M+\log_2(N)$만큼의 bit 크기를 갖는다.

그림 2는 롬으로 이루어진 분산연산부의 구조를 나타낸다. 이때 M은 계수의 bit수, N은 입력의 갯수, n은 출력의 bit수를 의미한다.

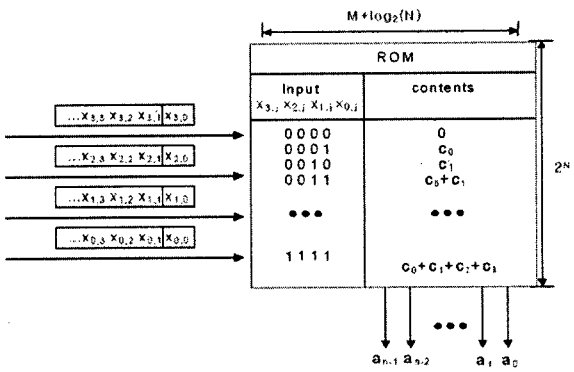

그림 2. 롬기반 분산연산 구조

4. 제안한 재구성 가능한 DCT/DWT의 구조

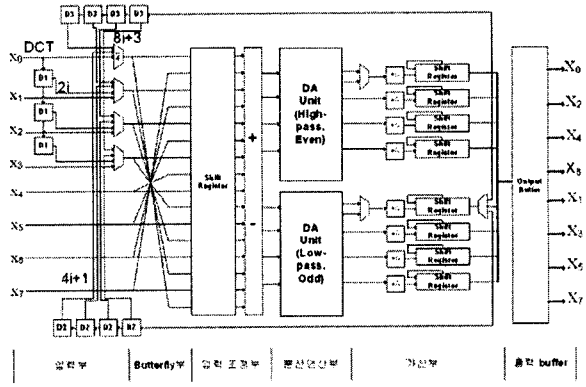

그림 3. 제안한 전체 시스템 구조

제안한 시스템은 그림 3과 같이 전체 6개의 부분으로 구성되어 있다. 입력부는 DCT연산과 3단계 DWT에 대한 입력을 선택하는 부분으로 4×1 MUX(Multiplexer)와 지연을 위한 레지스터로 구성될 수 있다. 8-point DCT 연산을 수행하기 위해서는 x_0부터 x_7까지 병렬로 입력되며, DWT 연산을 수행하기 위해서는 x_0만을 이용하여 입력값이 직렬로 입력하게 된다. 이때 4×1 MUX를 통해서 x_0부터 x_3까지는 각 단계에 맞는 DWT 입력을 선택하게 되며, x_4부터 x_7까지의 입력은 0을 입력하게 된다.

각 단계에서 DWT 연산 순서는 표 1과 같이 결정된다. 하나의 모듈을 가지고 3단계 DWT 연산을 수행하기 위해서 각 Cycle에서 중복되는 연산을 화살표와 같이 지연(Delay) 할 필요가 있다.

표 1. DWT 3단계의 각 Cycle에 대한 출력

입력(I)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	...
V (1단계 High)	V(0)		V(2)		V(4)		V(6)		V(8)		V(10)		V(12)		V(14)		V(16)	
X (2단계 High)	X(0)				X(4)				X(8)				X(12)				X(16)	
Z (3단계 High)	Z(0)								Z(8)								Z(16)	
Z' (3단계 Low)	Z'(0)								Z'(8)								Z'(16)	

연산순서 $D(k,i)$에 대해서 표 1을 정규화 시켜보면 식(9)을 얻을 수 있다. 식 (9)에 의해서 각 단계의 지연을 조정하면, 몇 단계의 연산이라도 원활하게 수행이 가능하도록 되어 있다.

$$D(k,i)=2^k i+(2^{k-1}-1) \quad (9)$$

여기서 $D(k,i)$는 단계 k에서 i번째 입력의 지연시간을 나타낸다.

입력 조정부는 DCT 연산에서 x_0+x_7과 x_0-x_7의 연산을 하기 위해 사용하는 부분으로써 그림 4와 같이 구성한다.

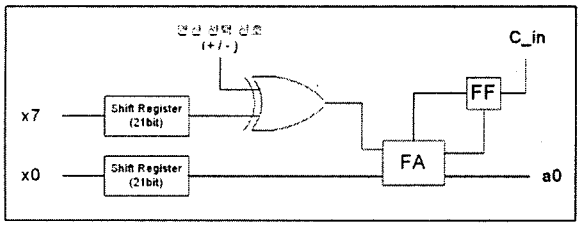

그림 4. 입력 조정부

초기에 C_in에 1을 입력하고 연산 선택신호를 1로 설정함으로써 x_0-x_7을 수행할수 있으며, C_in에 0을 입력하고 선택신호를 0으로 설정함으로써 x_0+x_7의 연산을 수행하게 된다. 분산연산부는 롬-기반 분산 연산부로 이루어져 있다.

가산부는 분산연산부에서 출력된 결과를 누적 함으로써 최종적인 결과값을 얻을 수 있다. 이때 DCT일 경우에는 X0부터 X7까지의 결과를 얻을수 있으며, DWT일 경우에는 각 단계의 고대역 통과필터의 값(V, X, Z)이 X0를 통해서 출력되며, 저대역 통과 필터의 값(Z')이 X1을 통해서 출력된다. 또한 저대역 통과 필터의 값은 k단계의 DWT 연산을 위해서 $k-1$단계에서 연산된 결과가 DeMUX (Demultiplexer)를 통해서 지연된다.

5. 실험 및 고찰

제안한 프로세서는 Verilog-HDL로 코딩하였으며, Altera의 Quartus II를 사용하여 시뮬레이션 하였다. 합성된 결과는 그림 5와 같다. 또한 Excalibur_ARM (EPXA10F1020C1)을 이용하여 구현하였고 동작속도는 47.85MHz이다.

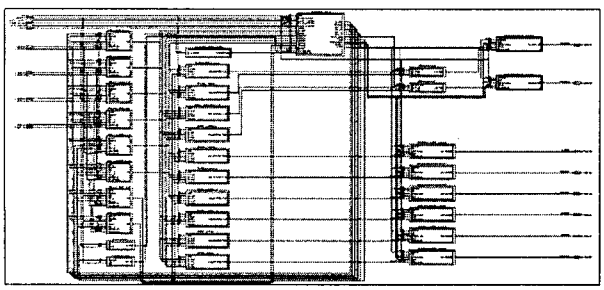

그림 5. 제안한 시스템의 합성구조

DCT 연산에서 8-point 연산의 경우 27clock이 필요하기 때문에, 512×512 영상 1프레임을 처리하는데 약 37ms가 소요되며, 256×256 영상 1프레임은 9ms에 처리할 수 있다. 그러므로, 512×512 영상은 초당 27프레임을 처리할 수 있다. 또한 3단계 DWT 연산에서는 256×256 영상 1프레임을 처리하는데 소요되는 시간은 152ms이며, 초당 7프레임을 처리할 수 있다.

6. 결론

본 논문에서는 하나의 모듈을 재구성하여 DCT/DWT 연산 처리를 수행하는 프로세서를 제안하였으며, 기존의 곱셈 연산 대신 롬-기반 분산연산을 사용함으로써 연산속도를 향상시킬 수 있었다.

향후 연구방향으로는 DWT 연산에서 발생하는 지연시간을 줄이는 방법과 동작속도를 빠르게 하기 위해서 ASIC으로 구현하는 것이 필요하다.

7. 참고논문

[1]Kondo. H, Kou. H, "Wavelet image compression using sub-block DCT", Proc Ninth IEEE international Conference on Networks 2001, pp 327-330, Oct. 2001.

[2]Nikolaidis. A, Pitas. I, "Asymptotically optimal detection for additive watermarking in the DCT and DWT domains", IEEE Transaction on Image Processing, Vol 12, Issue 5, pp 563-571, May. 2003.

[3]Fang Zhijun, Zhou Yuanhua, Zou Daowen, "A scalable video coding algorithm based DCT- DWT", Proc of the 3rd IEEE ISSPIT 2003, pp 247-249, Dec. 2003.

[4]R. Lang, E. Plesner, H. Schoder, and A. spray, "An efficient Systolic Architecture for the One-dimensional Wavelet Transform", in proc. SPIE Wavelet Applicaitoins, Orlando, FL, Vol. 2242, pp 925-935, Apr. 1994.

[5]Yu-Tai Chang, Chin-Liang Wang,"A New Fast DCT Algorithm and Its Systolic VLSI Implementation", IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II: ANALOG AND DIGITAL SIGNAL PROCESSING, Vol. 44, No. 11, pp 959-962, NOVEMBER. 1997.

[6]H. J. Lee, J. C. Liu, A. K. Chan, and C. K. chui, "Parallel Implementation of Wavelet Decomposition/ Reconstruction Algorithms", in proc. SPIE Wavelet Application Orlando, FL, Vol 2242, pp 248-259, Apr. 1994.

[7]Chang. T-S, Chen. C, Jen. C-W, "New distributed arithmetic algorithm and its application to IDCT", IEE Proceedings G-Circuits Devices and Systems, Vol 146, Issue 4, pp 159 - 163, Aug. 1999.

[8]Jong Tae Kim, Yong Hoon Lee, Tsuyoshi Isshiki, Hiroaki Kunieda, "Scalablel VLSI Architectures for Lattice Structure-based Discrete Wavelet Transform", IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II: ANALOG AND DIGITAL SIGNAL PROCESSING, Vol. 45, No. 8, pp 1031-1043, Aug. 1998.