MPEG-II AAC의 MDCT/IMDCT를 위한 벡터 프로세서 설계

이 강 현
조선대학교 전자정보통신공학부 멀티미디어 ASIC설계 실험실
http://vlsi.chosun.ac.kr
khrhee@vlsi.chosun.ac.kr

The Design of Vector Processor for MDCT/IMDCT of MPEG-II AAC

Kang Hyeon Rhee
School of Electronics, Information and Communication Eng.,
Multimedia ASIC Design Lab., Chosun University
Tel : (062) 230-7066 / Fax : (062) 233-1120
E-mail : khrhee@vlsi.chosun.ac.kr

Abstract

Currently, the most important technology is compression methods in the multimedia society.

In audio compression, the method using human auditory nervous property is used. This method using psychoacoustical model is applied to perceptual audio coding, because human's audibility is limited.

MPEG-II AAC(Advanced Audio Coding) is the most advanced coding scheme that is of benefit to high quality audio coding. The compression ratio is 1.4 times compared with MPEG-I layer-III.

In this paper, the vector processor for MDCT/IMDCT(Modified Discrete Cosine Transform /Inverse Modified Discrete Cosine Transform) of MPEG-II AAC is designed.

I. 서 론

MPEG-II AAC는 huffman coding, quantization and scaling, backward adaptive prediction, MDCT, gain control and hybrid filter bank (polyphase quadrature filter (IPQF)+IMDCT)로 구성되어 있다. MPEG-II AAC는 MPEG-I layer3에 비해 30% 압축 효율이 개선되었으며, 다 채널을 지원한다.[1,2]

MPEG-II AAC에서 연산량이 가장 많은 부분은 MDCT/IMDCT와 PQF/IPQF(Polyphase Quadrature Filter/Inverse Polyphase Quadrature Filter)라 할 수 있다. 그림 1은 AAC의 엔코더와 디코더이다. 그림 1의 Perceptual Model에서는 FFT를 연산을 하게되면 이득 제어기에서는 PQF을 수행하여 4개의 등간격 주파수 대역분할을 하게된다.[3,4] 이것을 필터뱅크(filterbank)에 넣어 서브밴드(subband) 분할한다.

본 연구는 반도체설계교육센터(IDEC)의 지원 장비 및 CAD툴에 의하여 수행된 연구입니다.

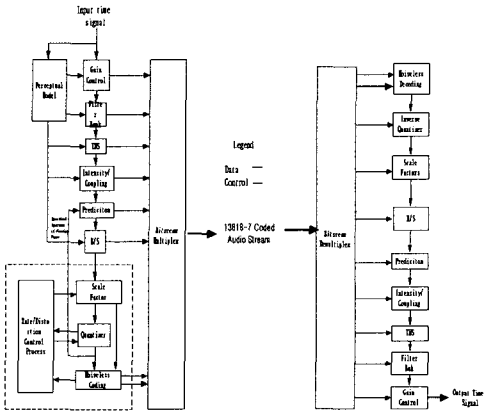

그림 1. AAC 엔코더/디코더 블록 다이어그램

II. MDCT/IMDCT의 최적화

MDCT는 시간영역 데이터를 주파수영역으로 변환하며, IMDCT는 역으로 주파수영역 데이터를 시간영역 데이터로 환원한다. MDCT의 장점으로는 임계 표본 값을 유지하기 때문에 중첩시간 원도우에서 50%

이상의 중첩에 의해서 시간영역의 에일리어싱(aliasing)이 제거된다.[5,6]

MDCT의 정의는 식 (1)과 같다.

$$X(i,k) = 2 \cdot \sum_{n=0}^{N-1} x(i,n) \cos\left(\frac{2\pi}{N}(n+n_0)\left(k+\frac{1}{2}\right)\right)$$

for $0 \le k < N/2$

x_{in} = 윈도된 입력
n = 샘플 인덱스
k = cosine계수인덱스
i = 블럭 인덱스
N = 윈도 길이
$n_0 = \left(\frac{N}{2}+1\right)\frac{1}{2}$ ····· (1)

IMDCT의 정의는 식 (2)와 같다.

$$X(i,n) = \frac{2}{N} \sum_{n=0}^{\frac{N}{2}-1} x(i,k) \cos\left(\frac{2\pi}{N}(n+n_0)\left(k+\frac{1}{2}\right)\right)$$

for $0 \le n < N$

x_{in} = 윈도된 입력
n = 샘플 인덱스
i = 윈도 인덱스
k = spectral계수인덱스
N = 윈도 길이
$n_0 = \left(\frac{N}{2}+1\right)\frac{1}{2}$ ····· (2)

식 (1)과 (2)를 비교하여 볼 때, 계수 인덱스 길이와 누산 후 2와 2/N을 곱하는 것 빼고는 차이가 없으므로, 식(1)을 최적화 하여 식(2)에 적용이 가능하다. MDCT의 장점에서 언급한 윈도우 함수의 반복 패턴을 보면 그림 2와 같다. 4종류의 윈도우 함수가 사용되고 36포인트 MDCT가 첫 번째로 오고, 12포인트 MDCT가 마지막 윈도우 함수로 사용된다.[7,8]

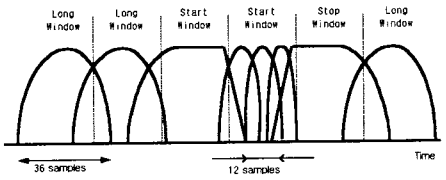

그림 2. 윈도우 함수의 반복 패턴

복소수영역에서 최적화 한 N/4점 MDCT의 흐름선도를 보면 그림 3과 같다. IMDCT도 같은 흐름선도를 갖으며, 단지 계수 값만 바꿔주면 된다. N개 샘플(Sample) MDCT의 계수 $p = \log_2 N$ 인 P 단계까지 분할되며, 또한 초기 연산에 필요한 계수는 ± 1이 되므로 연산에 필요한 계수 량은 $p = (\log_2 N) - 1$이 된다.

MPEG-II AAC의 경우 최대 2,048 MDCT까지 수행하여 필요한 계수의 개수는 2,048이지만 본 알고리즘을 사용하면 $p = (1og_2 2048) - 1$이 되어 하드웨어로 구현 시, 10개의 메모리 워드만 있으면 된다. 예를 들어, 2,048 포인트 IMDCT의 N=2,048, K=2,068이며 따라서 총 승산량은 1,024×10 = 10,400회가 된다.

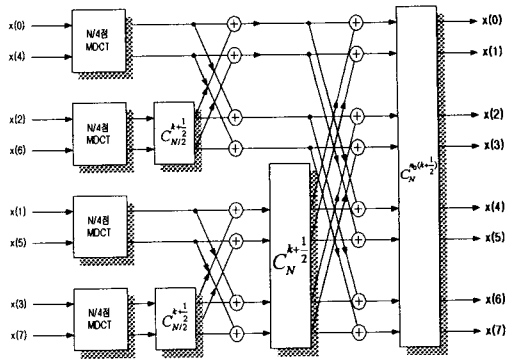

그림 3. N/4점 MDCT를 이용한 8점 MDCT

이것을 최적화 없이 계산할 때 승산량 4,194,304회와 비교할 때 약 400배정도 연산량이 감소함을 알 수 있으며 최적화 연산은 표 1과 같다.

표 1. K에 대한 최적화 연산

k =			
0.7071 + 0.7071i	0.9239 + 0.3827i	0.9808 - 0.1951i	0.9952 + 0.0980i
-0.7071 + 0.7071i	0.3827 + 0.9239i	0.8315 + 0.5556i	0.9569 + 0.2903i
-0.7071 - 0.7071i	-0.3827 + 0.9239i	0.5556 + 0.8315i	0.8819 + 0.4714i
0.7071 - 0.7071i	-0.9239 + 0.3827i	0.1951 + 0.9808i	0.7730 + 0.6344i
0.7071 + 0.7071i	-0.9239 - 0.3827i	-0.1951 + 0.9808i	0.6344 + 0.7730i
-0.7071 + 0.7071i	-0.3827 - 0.9239i	-0.5556 + 0.8315i	0.4714 + 0.8819i
-0.7071 - 0.7071i	0.3827 - 0.9239i	-0.8315 + 0.5556i	0.2903 + 0.9569i
0.7071 - 0.7071i	0.9239 - 0.3827i	-0.9808 + 0.1951i	0.0980 + 0.9952i
0.7071 + 0.7071i	0.9239 + 0.3827i	-0.9808 - 0.1915i	-0.0980 + 0.9952i
-0.7071 + 0.7071i	0.3827 + 0.9239i	-0.8315 - 0.5556i	-0.2903 + 0.9569i
-0.7071 - 0.7071i	-0.3827 + 0.9239i	-0.5556 - 0.8315i	-0.4714 + 0.8819i
0.7071 - 0.7071i	-0.9239 + 0.3827i	-0.1951 - 0.9808i	-0.6344 + 0.7730i
0.7071 + 0.7071i	-0.9239 - 0.3827i	0.1951 - 0.9808i	-0.7730 + 0.6344i
-0.7071 + 0.7071i	-0.3827 - 0.9239i	0.5556 - 0.8315i	-0.8819 + 0.4714i
-0.7071 - 0.7071i	0.3827 - 0.9239i	0.8315 - 0.5556i	-0.9569 + 0.2930i
0.7071 - 0.7071i	0.9239 - 0.3827i	0.9808 - 0.1951i	-0.9952 + 0.0890i

III. 마이크로프로그램방식의 MDCT 백터 프로세서의 설계

MDCT는 비교적 많은 누적연산을 필요로 하며, 또한 영상 데이터에 비해 더욱 높은 정밀도를 요구한다. 정밀도는 약 96dB가 요구된다. 따라서 본 시스템은 최대 2,048 포인트 MDCT에 대해 정밀도 96dB을 만족하도록 부동소수점 승산기와 가산기를 사용하며, 승산기는 지수에 6비트 가수부에 24비트, 그리고 가산기는 승산결과를 누적가산 할 수 있도록 6비트 지수부와 56비트 가수부를 갖는 부동소수점 가산기를 사용한다. 지수부는 다이내믹 레인지를 갖고 있으며, 가수부는 가수연산의 오차를 줄이기 위해 충분히 크게 설계하였으며, 가산기의 가수부가 56비트인 것은 MAC을 특별히 사용하지 않으므로 가산기가 승산 출력을 누적가산할 때 가수부 버림을 막기 위함이다. 연산의 속도를 최적화 하기 위해 승산기와 가산기는 모두 파이프라인

으로 설계하였다.

그림 4는 승산기의 가수부 승산을 하는 파이프라인 Radix-4 부스(Booth) 승산기의 가수부 승산에 사용되는 가산기의 가산구조를 보여준다. 병렬 부스 승산시 가수부 입력 비트 폭의 2배가되는 가산기가 필요하지만, 본 24비트 Radix-4 부스는 26비트 가산기로 가산 트리를 구성하며, 가산 트리의 마지막 노드만 31비트 가산기를 사용한다.

Radix-4 부스 승산기의 가산기 구조는 그림 4와 같다.

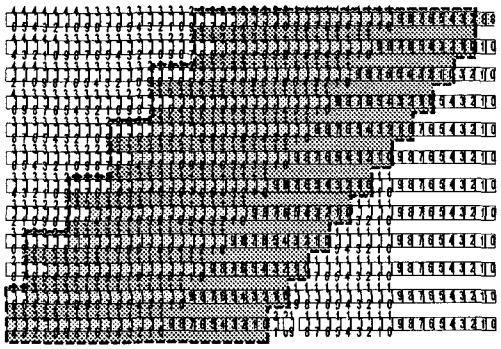

그림 4. Radix-4 부스 승산기의 가산기 구조

그림 5의 56bit 부동소수점 가산기는 캐리 선택 가산기(CSA:Carry Select Adder)를 사용한다.

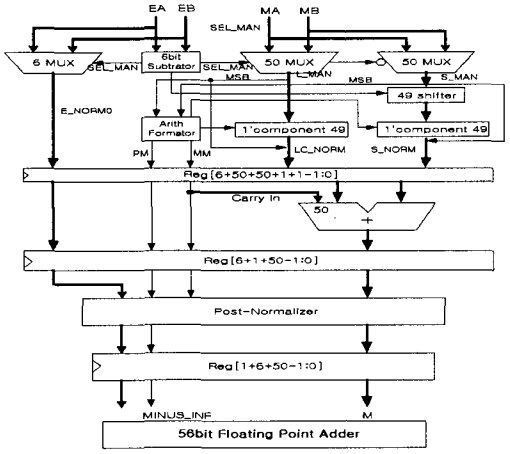

그림 5. 56비트 부동소수점 가산기

여기서 CSA방식을 사용하는 이유는 파이프라인 단계를 소비하지 않아도 되기 때문이다.

선 정규화기는 바이어스 되어 있는 부동소수점 수가 입력되는 것으로 가정하고, 선 정규화와 후 정규화에는 각각 46비트 배럴 쉬프터가 사용된다. 전체 시스템은 부동소수점 승산기 및 가산기, 벡터 레지스터를 대신하는 입출력 버퍼, 해저드 제어용 프로그래밍 지연 제어기, 그리고 마이크로 프로그램방식의 벡터 명령어 제어기로 구성된다. 벡터 레지스터를 대신하는 버퍼들은 외부의 동기식 RAM과 인터페이스 되어 동작하므로 연산에 오버헤드가 없게 하였다.

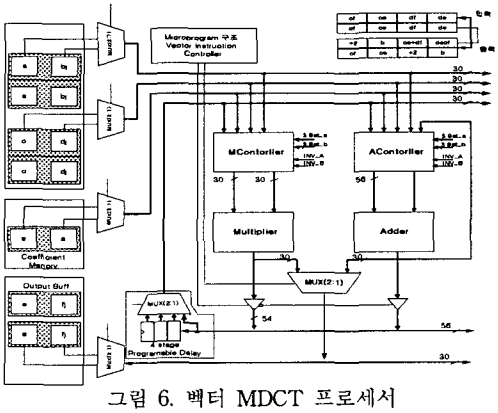

그림 6. 벡터 MDCT 프로세서

그림 7은 마이크로 프로그램방식의 벡터 명령어 제어기이다

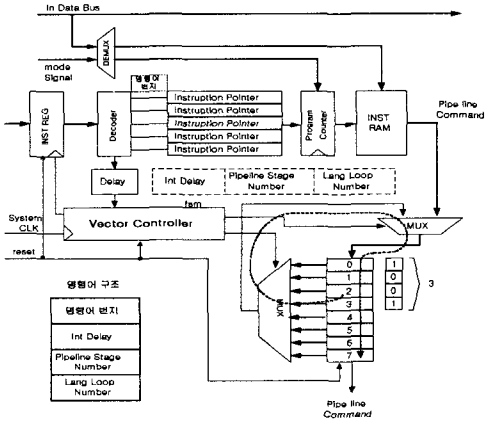

그림 7. 파이프라인 구조의 벡터 명령어 제어기

벡터 명령어는 승산기와 가산기로 수행할 기본 연산 요소 즉, 마이크로 프로그램 명령을 정의(스칼라 연산)하고 이러한 구조의 특징은 사용자가 수행하고자 하는 연산은 해저드를 고려하여 최적의 마이크로 프로그램 명령을 정의하고, 이 기본 명령에 기초한 응용프로그램을 동작되게 함으로서, 최적의 프로세서 성능을 발휘할 수 있게 함과 동시에 소프트웨어처럼 유연한 프로세서를 구현할 수 있다.

본 설계에서 구현된 칩은 초기화 시에 사용자가 외부 ROM을 사용하여 칩의 벡터 명령어를 정의하고,

동작모드에서 이것을 조합한 응용프로그램이 실행될 수 있도록 하였다. 제어기는 명령어 RAM, 명령어 포인터, 벡터 명령어용 환형 버퍼와 쉬프터로 구성되었다.

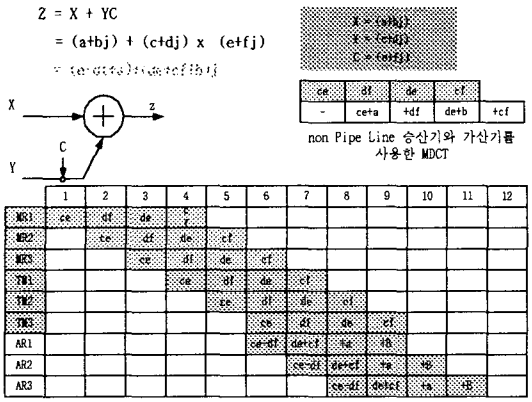

그림 8. MDCT/IMDCT 연산요소의 스케줄

그림 8은 MDCT와 IMDCT에서 사용하는 연산요소의 파이프라인 해저드를 제거한 파이프라인 스케줄러인데, 부동소수점 승산기 및 가산기의 모든 입력은 클록의 상승 에지 에서 동작하며, 출력은 하강 에지에서 동작하게 된다.

프로그래머블 지연이 해저드제어를 하며, 2개의 복소수 입력과 1개의 복소수 계수를 곱하여 가산하는 연산인 경우 4개의 사이클을 소비하여 실수 계산값과, 허수 계산값 모두를 얻어낸다.

IV. 결 론

본 논문의 설계에서는 MPEG-II AAC에서 사용하는 MDCT/IMDCT를 최적화 하여, 최적화 전보다 약 0.25% 연산으로 처리 가능하게 하였으며, 설계된 프로세서가 MPEG-II AAC의 기타 다른 연산까지 충분히 처리할 수 있도록 30비트 부동소수점 승산기와 56비트 부동소수점 가산기로 초고속 프로세서를 설계하였다.

설계된 칩은 완전한 파이프라인으로 동작하며, 파이프라인 명령과 파이프라인 스케줄링을 모두 칩의 사용자가 정의하므로 설계자의 입장에서 제어기 설계 자체가 매우 쉬우며, 또한 사용자 입장에서는 명령어를 호환시켜 재개발 없이 응용프로그램을 바이너리 호환하여 사용할 수 있다.

한편 이러한 명령어 제어기를 좀더 보완할 경우 타 프로세서와 완벽한 바이너리 호환 능력을 갖출 수 있으며, JAVA 프로세서와 같은 가상머신을 탑재한 프로세서 설계에 적용할 수 있다.

참 고 문 헌

[1] J.Princen, A.Johnson, A.Bradley: "Subband/Transform Coding Using Filter Bank Designs Based on Time Domain Aliasing Cancellation," Proc, of the ICASSP 1987, pp. 2161-2164.

[2] "Presented at the 101st Convention 1996 November 8-11 Los Angeles, California," AN AUDIO ENGINEERING SOCIETY PREPRINT. p28

[3] ISO/IEC JTC1/SC29/WG11 N1650, "IS 13818-7(MPEG-2 Advanced Audio Coding:AAC)" April 1997, p49, p77.

[4] Mark Kahrs, Karlheinz Brandenburg, "APPLICATIONS OF DIGITAL SIGNAL PROCESSING TO AUDIO AND ACOUSTICS," 1998 by Kluwer Academic Publishers.

[5] "The Digital Signal Processing HAND BOOK" Edited by vijay k. Madisetti , Douglas B. Williams, CRC PRESS, IEEE PRESS

[6] T. Mochizuki, "Perfect Reconstruction Conditions for Adaptive Blocksize MDCT," Trans. IEICE, vol. E77-A, no. 5, pp. 894-899, May 1994.

[7] M. Vetterli et al., "Perfect Reconstruction FIR Filter Banks: Some Properties and Factorizations," IEEE Trans. ASSP vol. 37, pp. 1057-1071,

[8] P. P. Vaidyanathan, "Multirate Digital Filter, Filter Banks, Polyphase Networks, and Applications: A Tutorial," Proc IEEE vol. 78, no. 1, pp.56-93,