2000年度 大韓電子工學會 夏季綜合學術大會 論文集 第 23 卷 第 1 號 2000/6

VHDL을 이용한 MPEG-2 AAC 복호화기 필터뱅크의 구현

우 광 희(禹 光 熙), 차 형 태(車 亨 泰)
숭실대학교 전자공학과
전화 : (02) 826-9063 / 팩스 : (02) 820-0711

Implementation of filterbank for MPEG-2 AAC decoder with VHDL

Kwanghee Woo, Hyungtai Cha
Soongsil University, School of Electronic Engineering,
E-mail : com3com4@mmslab.ssu.ac.kr

Abstract

In this paper, we present the implementation of filterbank for MPEG-2 Advanced Audio Coding (AAC) decoder with VHDL. The filterbank of AAC employs a technique called time-domain aliasing cancellation (TDAC). In order to make the algorithm more efficiently, we decompose and reorganize the filterbank algorithm for the high speed decoding process and lower computational cost. And we make this filterbank algorithm to be used with other modules of AAC decoder in parallel processing.

I. 서론

1994년, MPEG-2 오디오 표준화위원회는 멀티채널을 지원하기 위한 고 압축을 지원하는 새로운 압축 알고리즘 표준화 작업에 들어갔는데, 기존의 MPEG-1 오디오와의 호환을 포기하고 MPEG-2 오디오 NBC (Non-Backward Compatible)오디오 표준을 제안하였다. 이 새로운 오디오 부호화 알고리즘은 MPEG- 2 Advanced Audio Coding(MPEG-2 AAC)으로 표준화되었다[4][1]. MPEG-2 AAC는 1998년 12월에 표준화된 MPEG-4 오디오의 T/F코더에 TwinVQ, BSAC와 함께 메인 커널로 채택되었다[2].

MPEG-2 AAC 알고리즘을 디지털 방송 및 멀티미디어 시스템에 적용하기 위해서는 실시간 처리 가능한 부호화 및 복호화 시스템이 개발되어야 한다.

본 논문에서는 MPEG-2 AAC 복호화기의 필터뱅크 알고리즘을 칩 설계에 적합하도록 최적화하고, VHDL (Very High Speed Integrated Circuit Hardware Description Language)을 이용하여 설계하였다.

II. MPEG-2 AAC의 기본 알고리즘

그림 1은 MPEG-2 AAC의 전체 복호화 과정을 나타내었다.

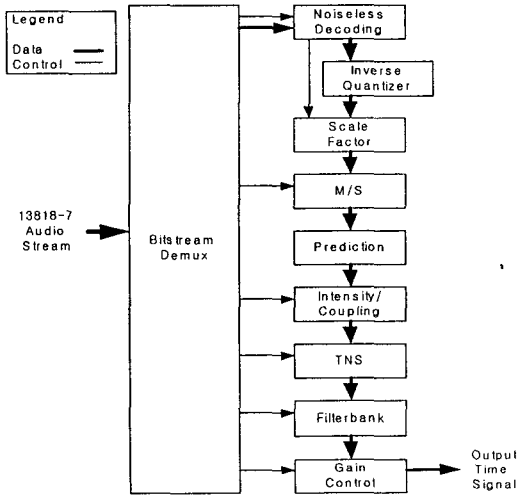

그림 1 . MPEG-2 AAC복호화기 블럭도
Fig. 1 . Block Diagram of MPEG-2 AAC Decoder

복호화 과정은 AAC 부호화기에 의해 부호화된 비트열을 분석하여 허프만 복호화, 역양자화를 수행한다. 역 양자화된 샘플을 허프만 복호화된 스케일팩터로 스케일링하고 M/S 스테레오를 적용한다. 예측기를 거쳐 인텐시티/커플링을 수행하고 시간영역잡음 변형을 수행한 후 필터뱅크를 거쳐 시간영역의 신호로 변환되는

과정이다[1].

MPEG-2 AAC 복호화기의 필터뱅크는 주파수 영역 입력신호를 시간영역의 샘플로 변환하고, 윈도우와 오버랩애드(Overlap Add)를 수행한다[1].

III. AAC 복호화기 필터뱅크 최적화

복호화기의 하드웨어 구현을 위해 우선 복호화기 각 툴의 연산량에 따른 복잡도를 계산해야 한다. 1998년 MPEG 오디오 서브그룹에서는 AAC의 세 개의 프로필에 대한 각 툴의 연산량의 비교에 따른 소프트웨어와 하드웨어 구현을 위한 테스트를 수행하였다[3]. 표 1에서와 같이 필터뱅크는 AAC 복호화기의 전체 연산량에서 가장 많은 비중을 차지한다.

표 1. LC 프로필의 연산량 비율[3]

Table 1. Instruction Complexity(LC profile)

	1 채널	5 채널	비율
허프만 복호화	13,657	68,285	32.5
역양자화	1,708	8,540	4.1
M/S		1,708	0.8
커플링		11,546	5.5
TNS	4,065	20,325	9.7
필터뱅크	19,968	99,840	47.5
합계	39,398	210,244	100.0

MPEG-2 AAC 시스템의 가장 기본적인 요소로서 필터뱅크는 시간 시간영역과 내부적인 시간 주파수 영역으로 상호 변환을 수행한다. 부호화기에서 MDCT (Modified Discrete Cosine Transform)에 의해 변환되고, 복호화기에서는 IMDCT(Inverse MDCT)에 의해 역변환 된다. MDCT와 IMDCT는 Dolby AC-3에서 사용된 Time-Domain Aliasing Cancellation(TDAC) 기법을 사용한다[7]. 필터뱅크에 사용되는 IMDCT의 수식은 식(1)과 같다[1].

$$x_{i,n} = \frac{2}{N}\sum_{k=0}^{N/2-1} X_{i,k}\cos\left(\frac{2\pi}{N}(n+n_0)\left(k+\frac{1}{2}\right)\right),\ 0 \le n < N \quad (1)$$

여기서,

x = 시간영역 신호, n = 샘플의 인덱스,

X = 스펙트럼 계수, k = 스펙트럼 계수 인덱스

i = 블럭 인덱스, N = 윈도우의 길이(2048 또는 256)

n_0 (N/2+1)/2 이다.

식 (1)에서 시간영역의 한 샘플을 구하기 위해 X와 cos계수를 N/2번 곱하고 더하는 계산을 수행하여야 한다. 이러한 샘플을 N개 계산하는데는 $N^2/2$번의 계산이 수행된다. 본 논문에서는 N/2포인트 IMDCT를 N/4포인트 IFFT로 구현하였다. 식(1)을 IFFT를 이용한 방법으로 나타내면 식(2)와 같다[6][7].

$$x'[n] = \left[\sum_{k=0}^{N/4-1}\left\{X'[k]\cdot e^{j\frac{2\pi}{N}\left(k+\frac{1}{8}\right)}\right\}e^{j\frac{2\pi}{N/4}nk}\right]e^{j\frac{2\pi}{N}\left(n+\frac{1}{8}\right)} \quad (2)$$

여기서,

$$X'[k] = -X[2k] + jX[N/2-2k-1]$$

$$x[\frac{3N}{4}-1-2n] = real(x'[n]),\quad x[\frac{3N}{4}+2n] = real(x'[n]),$$

$$x[\frac{N}{4}+2n] = img(x'[n]),\quad x[\frac{N}{4}-1-2n] = -img(x'[n]),$$

$$x[\frac{N}{2}-1-2n] = real(x'[\frac{N}{8}+n]),\quad x[2n] = -real(x'[\frac{N}{8}+n]),$$

$$x[\frac{N}{2}+2n] = img(x'[\frac{N}{8}+n]),\quad x[N-1-2n] = img(x'[\frac{N}{8}+n])$$

식(2)에서 $\sum_{k=0}^{N/4-1}\{\cdot\}e^{j\frac{2\pi}{N/4}nk}$ 는 N/4-포인트 IFFT임을 알 수 있다. 즉, IFFT 입력 단에 입력 스펙트럼 계수를 전처리하고 출력 단에서 후처리를 수행하여야 한다. 이 과정은 그림 2와 같이 정리할 수 있다.

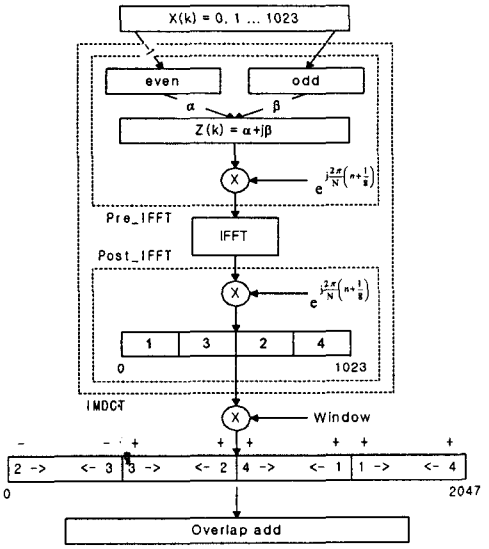

그림 2. AAC 복호화기 필터뱅크 수행과정

Fig. 2. Filterbank Processing of AAC Decoder

전처리 과정에서는 입력 샘플의 짝·홀수의 순서로 실·허수의 복소수로 처리하여 복소수 곱셈을 하고, 후처리 과정에서는 IFFT출력 결과에 복소수 곱셈을 하여 N/2개의 샘플을 디인터리브(de-interleave)하여 샘플의 순서를 바꾸는 작업을 하게 된다. IMDCT의

출력 결과를 윈도우를 적용하여 이전 샘플과 50% 오버랩하여 시간영역 신호를 출력한다.

IV. 필터뱅크의 하드웨어 구현

본 논문에서는 IMDCT의 연산을 고속화하기 위하여 그림 3과 같이 radix 2 DIF(Decimation in Frequency) 버터플라이 모듈을 선택하였다.

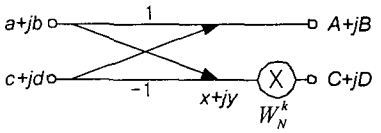

그림 3. DIF IFFT 버터플라이의 구조
Fig. 3. Butterfly of DIF IFFT

radix 2 버터플라이 모듈은 입력과 출력의 인덱스의 순서가 VHDL로 구현하기에 적합한 알고리즘이다. 또한 DIF 버터플라이 구조는 4번의 곱셈이 수행되는 동안 덧셈과 뺄셈을 파이프라인 구조로 처리하기에 적합한 알고리즘이다.

그림 3과 같은 버터플라이를 구현하기 위해 실수와 허수를 구성하는 각각의 RAM을 두어 한 클럭에 실수와 허수를 읽고 쓸 수 있게 구성하였다. 연산부는 24비트 곱셈기, 덧셈기, 뺄셈기를 각각 구성하여 각각의 연산기가 동시에 수행 될 수 있도록 구성하였다.

표 2는 그림 1의 radix 2 DIF 버터플라이 하나를 수행하기 위한 클럭별 수행 순서를 나타내었다. 표 2의 결과로 하나의 버터플라이를 수행하는데 8클럭이 소요되며, 다음 버터플라이는 5번째 클럭에서 시작하여 파이프라인으로 구성하였다. 따라서, 전체 IFFT의 버터플라이는 평균 4클럭에 수행하였다.

표 2. 버터플라이 수행 순서
Table 2. Clock Table of Butterfly Processing

Clk	Add	Sub	Mul	RAMr	RAMi
1				R(a)	R(b)
2				R(c)	R(d)
3	A=a+c	x=a-c			
4		y=b-d	x×Wr		
5	B=b+d		y×Wi		
6		C	y×Wi		
7			x×Wr	W(A)	W(B)
8	D			W(C)	W(D)

IFFT는 입력과 출력의 주소 인덱스가 서로 일치하지 않는다. 그림 4에 8-포인트 IFFT의 입력과 출력을 예로 나타내었다.

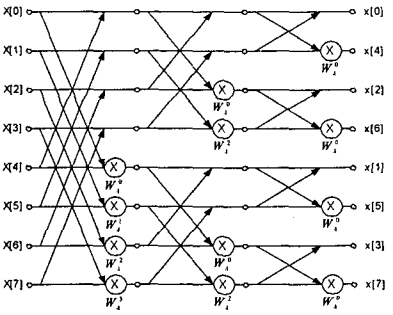

그림 4. 8-포인트 IFFT 버터플라이
Fig. 4. 8-Point IFFT Butterfly

8-포인트 IFFT를 예로 들면 출력의 순서는 입력 인덱스의 최상위 수 7을 표현할 수 있는 2진수 3비트로 표현되는데, 출력 인덱스의 MSB와 LSB가 바뀌어 (bit-reverse) 표현된다. 이러한 과정은 재배열(reordering)이 필요한데, VHDL은 비트단위 인덱스처리가 가능하여 재배열 없이 후처리 과정의 입력 인덱스의 MSB와 LSB를 바꾸어 사용하였다. W_N의 값은 ROM에 미리 정의 해 두었으며 sin(·), cos(·)함수의 주기적인 특성으로 반주기만을 정의하여 나머지 반주기는 인덱스를 감소 시켜 사용하였다.

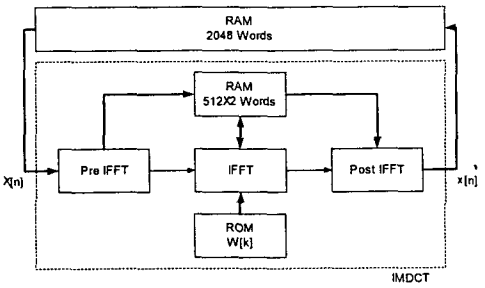

그림 5. AAC IMDCT 하드웨어 블록도
Fig. 5. H/W Block Diagram of AAC IMDCT

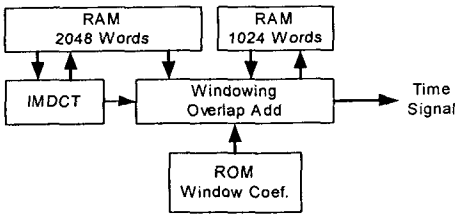

그림 6. AAC의 필터뱅크 하드웨어 블록도
Fig. 6. H/W Block Diagram of AAC Filterbank

그림 5에 IMDCT의 블록도를 나타내었고 그림 6에 IMDCT의 출력을 윈도우와 오버랩애드를 수행하는 전체 필터뱅크의 블록도를 나타내었다.

V. 실험 및 결과

본 논문에서는 MPEG-2 AAC 복호화기에 사용되는 고속 필터뱅크를 VHDL을 이용하여 설계하였으며, Synopsys를 이용하여 합성하였다. 그림 7에 Synopsys를 이용하여 게이트 수준 최적화를 수행한 합성도를 나타내었다.

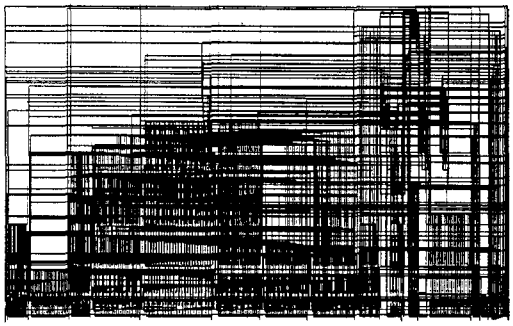

그림 7. 필터뱅크의 Synopsys 합성도
Fig. 7. Synopsys Synthesis of Filterbank

IFFT를 수행하기 위한 전처리과정과 후처리 과정을 하나의 모듈로 구성하여 입·출력만 바꾸어 구현하여 소요되는 게이트 수를 줄였으며, 내부 레지스터를 줄이기 위해 IFFT에서 사용된 레지스터를 전처리와 후처리에 재사용 하여 효율을 더 높였다. 모든 메모리와 연산부를 AAC 복호화기의 다른 툴들과 공유하도록 구성하였다.

표 3은 필터뱅크의 각 기능별 클럭 수를 나타내었다.

표 3. 필터뱅크 수행 속도
Table 3. Processing speed of the filterbank

	전처리	IFFT	후처리	Window Overlap -add	계
Clk	2,052	9,224	2,052	2,052	15,380
비율	13.3%	60%	13.3%	13.3%	100%

VI. 결론 및 추후 연구

본 논문에서는 MPEG-2 AAC 복호화기의 필터뱅크를 VHDL을 이용하여 구현하였다. AAC 복호화기전체 시스템에서 실시간 구현을 위해 필터뱅크의 계산속도를 IFFT를 이용하여 향상시켰다.

본 논문에서 구현한 필터뱅크는 6채널 48kHz 샘플링 주파수의 복호화기에 사용할 수 있다. 실시간 동작을 검증하기 위해 복호화기 전체 시스템의 수행 시간과 필터뱅크의 수행시간을 비교해 클럭주파수를 결정할 수 있다.

5.1 채널 48kHz 샘플링 율에서 한 프레임은 1024*6 샘플씩 처리되며 21.3 ms에 해당한다. 채널 당 21.3/6 = 3.55 ms 이내에 처리할 수 있어야 실시간 복호화가 가능하다.

표 1을 참조하여 전체 복호화기 수행시간에서 필터뱅크가 수행되는 시간을 최소 35%정도로 하면 필터뱅크는 1.2425 ms이내에 수행되어야 한다. 표 3의 결과로, 본 논문에서 구현된 필터뱅크 전체의 수행 클럭은 15,380 클럭동안 수행되었으며, 1.2425 ms이내에 수행되기 위해서 13 MHz의 클럭 주파수를 사용 할 수 있다.

AAC 복호화기 툴들 중 필터뱅크 다음으로 연산량을 많이 차지하는 허프만 복호화기를 필터뱅크와 동시에 수행 될 수 있도록 설계하면 클럭 주파수를 더 낮게 사용 할 수 있다.

Reference

[1] ISO/IEC 13818-7, "Generic Coding of Moving Pictures and Associated Audio Information - Part 7 : Advanced Audio Coding", 1997

[2] ISO/IEC 14496-3, "Information Technology - Coding of Audiovisual Objects - Part 3 : Audio, Subpart 4 : T/F Coding", 1998

[3] ISO/IEC JTC1/SC29/WG11 N2005, "Revised Report on Complexity of MPEG-2 AAC Tools" ", Feb. 1998

[4] M. Bosi, "Overview of MPEG Audio : Current and Future Standards for Low-Bit-Rate Audio Coding", J. AES , Vol. 45, No. 1/2, Jan/Feb. 1997, pp. 4-21

[5] M. Bosi, "ISO/IEC MPEG-2 Advanced Audio Coding", J. AES , Vol. 45, No. 10, Oct. 1997, pp. 789-814

[6] Rolf Gluth, "Regular FFT-Related Transform Kernels for DCT/DST-Based Polyphase Filter banks", ICASSP, vol.3, 1991, pp. 2205-8

[7] ATSC, "Digital Audio Compression Standard (AC-3)", Dec. 1995