1999 年 度 大 韓 電 子 工 學 會 夏 季 綜 合 學 術 大 會 論 文 集 第 22 卷 第 1 號 99/6

# VLSI 를 이용한 MPEG-2 AAC 복호화기 설계

이근섭, 정남훈, 방경호, 윤대희
연세대학교 전자공학과
전화 : (02) 361-2863 / 팩스 : (02) 312-4584

# VLSI Design of MPEG-2 AAC Decoder

**Keun-Sup Lee, Nam-Hun Jeong, Kyoung-Ho Bang, and Dae-Hee Youn**
**ASSP Lab., Electronic Eng., Yonsei Univ.**
**Email :** taraji@cyclon.yonsei.ac.kr

## Abstract

This paper presents a real-time MPEG-2 AAC decoding system, which can decode 2-channel main profile MPEG-2 AAC bitstream. The proposed system supports all decoding tools except for coupling channel tool, and provides sampling rates of 32, 44.1, 48 kHz. The system consists of a simple programmable DSP core and two hardwired logic modules that perform Huffman decoding and prediction for real-time implementation.

## I. 서론

국제 표준화 기구 산하 동영상 전문가 그룹은 MPEG-1 오디오 표준안과의 호환 요건 없이 뛰어난 음질을 제공하는 MPEG-2 AAC 표준안을 제정하였다. MPEG-2 AAC 알고리듬은 낮은 비트율에서도 방송에 적합한 뛰어난 음질을 얻기 위해 고해상도 필터 뱅크, 예측, 허프만 부호화 등의 기술들을 집약시켜 놓았다 [1][2].

MPEG-2 AAC 알고리듬을 방송이나 멀티미디어 시스템 등에 이용하기 위해서는 실시간 처리가 가능한 부호화/복호화 시스템을 구현하는 것이 필요하다. VLSI 에 의한 ASIC 기술은 복잡하고 방대한 시스템을 소형화하는 것을 가능하게 만들었고, 이미 많은 오디오 부호화/복호화 시스템이 ASIC 기술을 이용하여 단일 칩으로 개발, 상용화되었다.

본 논문에서는 MPEG-2 AAC 표준 비트열로부터 오디오 신호를 실시간으로 복호화하는 하드웨어 시스템을 설계하였다. 복호화 시스템은 크게 고정 소수점 DSP 프로세서 코어 모듈과 특정한 연산을 독립적으로 반복 수행하는 전용 하드웨어 모듈로 이루어지며, 각각의 모듈은 VHDL 로 설계되었다. 본 논문에서 사용된 프로세서 코어는 오디오 부호화/복호화 시스템에 적합하도록 설계되었으며, 이미 다른 오디오 부호화/복호화 시스템의 구현을 통해 기능적, 물리적으로 검증되었다. 그리고 방대한 연산량으로 인한 프로세서 코어의 부담을 덜고 실시간으로 동작하는 효과적인 시스템을 구현하기 위해 각 연산들의 관계와 소요 시간 등을 고려하여 허프만(Huffman) 복호화 과정과 예측 과정을 독립된 하드웨어 모듈로 나누고, 최적화된 구조를 제안하였다.

구현된 복호화기는 2 채널의 오디오 신호를 실시간으로 복호화하며 MPEG-2 AAC 표준안에서 제안하고 있는 기법 중 커플링 채널 기법을 제외한 모든 기능을 지원한다. 또한 MPEG-2 AAC 의 세가지 프로필 중 메인 프로필을 지원하고 표본화 주파수는 32, 44.1, 48 kHz 만을 지원한다.

논문의 구성은 다음과 같다. 2 장에서는 복호화 알고리듬을 실시간으로 구현하기 위한 알고리듬의 최적화에 대해 설명하였고, 3 장에서는 이를 바탕으로 한 효율적인 복호화기의 구조에 대해 설명하였다. 4 장에서는 설계된 복호화기의 실시간 구현 검증과 알고리듬의 검증을 통해 시스템의 성능을 평가하였다. 마지막으로 5 장에서는 결론으로 끝을 맺었다.

## II. 복호화 알고리듬의 실시간 구현

복호화가 실시간으로 처리되기 위해서는 필터 뱅크, 예측, 무손실 부호화, TNS, 그리고 채널간 부호화의 과정들이 하나의 프레임에 해당되는 시간 내에 이루어져

야 한다. 따라서 가능한 적은 연산량으로 복호화를 구현할 수 있는 알고리듬의 최적화가 반드시 필요하다.

시스템 구현에 사용된 프로그래머블 DSP 프로세서 코어는 지수 연산을 하나의 명령어로 지원하지 않으므로, 지수 연산을 사용하는 역 양자화 과정과 크기 인자 과정의 경우 고정 소수점 DSP 프로세서에서 수행될 수 있는 지수 연산 알고리듬이 필요하다. 일반적으로 고정 소수점 DSP 프로세서로 지수 연산을 구현하면 많은 연산량이 필요할 뿐만 아니라 여기서 발생하는 고정 소수점 오차도 크므로 이 두가지를 모두 만족시킬 수 있도록 최적화가 이루어져야 한다.

## 2.1 역 양자화 과정 최적화

역 양자화 과정은 비선형 양자화기로 양자화되어 정수 형태로 표현되어있는 스펙트럼 계수들을 원래의 실수 값으로 되돌리는 과정으로[1] 식은 다음과 같다.

$$x\_invquant = Sign(x\_quant)\cdot|x\_quant|^{\frac{4}{3}} \quad \cdots\cdots\cdots (1)$$

본 논문에서는 미리 계산 결과를 테이블로 만들어 놓고 입력 값에 따라 테이블에서 결과 값을 찾는(table look-up) 방법을 사용하였다. 그러나 테이블을 사용하므로 입력 값의 범위만큼 메모리가 요구되는 단점이 있다. MPEG-2 AAC 에서 역 양자화 과정의 입력 값 $x\_quant$ 의 절대값 범위는 $0 \le |x\_quant| \le 8191$ 이므로 [1] 필요한 테이블의 크기는 8192 이다. 이정도의 테이블 크기는 실제로 메모리를 사용해서 구현하기에는 무리가 있으므로 테이블 크기를 줄일 수 있는 방법이 요구된다. 본 논문에서는 다음과 같은 방법을 사용하여 작은 크기의 테이블로 역 양자화 과정을 수행할 수 있도록 하였다.

$$\begin{aligned} |x\_quant|^{\frac{4}{3}} &= |(x\_quant/8)\times 8|^{\frac{4}{3}} \\ &= |x\_quant/8|^{\frac{4}{3}} \times 8^{\frac{4}{3}} \quad \cdots\cdots\cdots (2) \\ &= |x\_quant'|^{\frac{4}{3}} \times 16 \end{aligned}$$

단, $x\_quant' = x\_quant/8$ 이다.

위와 같은 방법을 사용하면 범위가 큰 $x\_quant$ 를 범위가 작은 $x\_quant'$ 로 대신할 수 있다. $x\_quant'$ 는 $x\_quant$ 를 8 로 나눈 값이므로 값의 범위가 1/8 로 줄어든다. 따라서 $x\_quant'$ 의 절대값 범위는 $0 \le |x\_quant'| \le 1023$ 가 된다. 따라서 1024 크기의 테이블로도 역양자화 과정을 수행할 수 있다. 그러나 테이블에서 값을 찾기 위해서는 $x\_quant'$ 값이 정수여야 하므로 실제로 8 로 나누는 연산은 3 비트만큼 오른쪽으로 쉬프트 하는 연산으로 대신하게 된다. 이 경우 쉬프트 연산을 해서 잃게 되는 비트만큼 오차가 발생하는 것을 피할 수 없다.

이러한 오차를 줄이기 위해서 선형 보간법을 사용하였다. 쉬프트 연산에서 잃게 되는 3 비트는 결국 $x\_quant$ 를 8 로 나누었을 때의 소수 부분이다. 따라서 이 3 비트를 사용하여 선형 보간법을 수행하면 오차를 줄일 수 있다. 본 논문에서는 위와 같은 과정을 한번 더 수행하여 필요한 테이블 크기를 128 로 줄였다.

## 2.2 크기 인자 과정 최적화

크기 인자 과정은 역 양자화 과정을 거친 스펙트럼 계수에 크기 인자에 해당하는 이득을 곱해주는 과정이다[1]. 정수 형태의 크기 인자로부터 이득을 구하는 방법은 다음과 같다.

$$Gain = 2^{0.25\times(scalefactor - SF\_OFFSET)} \quad \cdots\cdots\cdots (3)$$

위 식에서 $SF\_OFFSET = 100$ 이다.

이 식도 또한 지수 연산이므로 이것을 구현할 수 있는 특별한 알고리듬이 필요하다. 이 지수 연산은 밑이 2 이므로 쉬프트 연산과 약간의 테이블로 쉽게 구현이 가능하다. 구현 방법은 다음과 같다.

0.25 를 곱하는 것은 4 로 나누는 것과 같으므로 결국 소수점이 왼쪽으로 2 비트 만큼 이동하는 결과를 낳는다. 따라서 20 비트의 $(Scalefactor - SF\_OFFSET)$ 값에 0.25 를 곱하면 소수점이 이동하여 18 비트의 정수 부분 $SF_i$ 와 2 비트의 소수 부분 $SF_f$ 로 나누어진다. 그런데 $SF_f$ 는 2 비트이므로 가질 수 있는 값이 4 가지이다. 따라서 $2^{SF_f}$ 의 4 가지 결과 값을 미리 계산하여 테이블로 만들어 놓으면 테이블 값을 찾는 것으로 빠른 연산이 가능하다. 또한 $2^{SF_i}$ 는 2 의 정수승 연산이므로 쉬프트 연산으로의 대체가 가능하다. 따라서 크기 인자 이득은 $2^{SF_f}$ 값을 테이블에서 찾은 후 그 값을 $SF_i$ 만큼 쉬프트하는 방법을 사용하여 구할 수 있다.

# III. 효율적인 복호화기의 설계

최적화 과정을 거친 복호화 알고리듬은 VLSI 를 통해 하나의 시스템으로 구현된다. MPEG-2 AAC 오디오 복호화기를 하나의 프로세서 코어를 사용하여 구현하기 위해서는 알고리듬의 복잡성과 많은 연산량으로 높은 성능의 프로세서가 필요하다. 그러나 효율적인 시스템을 구현하기 위해서는 복호화 과정 중 제어 중심의 과정을 프로세서 코어에 할당하고 연산 중심의 과정을 하드와이어드 로직(hardwired logic)으로 설계된 모듈에 할당하는 것이 유리하다[3]. 본 논문에서는 위와 같은 사항들을 고려하여 전체 시스템을 몇 개의 모듈로 나누어 구현하였다. 시스템의 복잡도와 효율성을 고려하여 프로세서 코어의 성능이 허락하는 한도

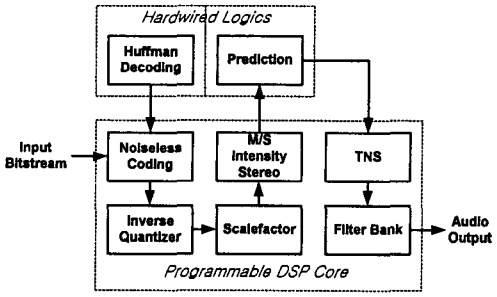


그림 1. MPEG-2 AAC 복호화기의 전체 구조

내에서 대부분의 복호화 과정들을 프로세서 코어를 사용하여 구현하였고, 연산량과 데이터의 이동, 그리고 사용되는 자원 등을 고려해서 허프만 복호화 과정과 예측 과정을 외부의 모듈로 분리하여 구현하였다.

설계된 MPEG-2 AAC 오디오 복호화기의 구조를 그림 1 에 나타내었다.

구현된 허프만 복호화 모듈과 예측 모듈에 대한 설명은 다음과 같다. 프로세서 코어에 대해서는 참고문헌 [3]에 자세히 설명되어있다.

## 3.1 허프만 복호화 모듈

허프만 복호화 과정은 비트열에 포함되어있는 허프만 코드 워드에 대한 허프만 코드 인덱스를 구하는 과정이다[1]. 먼저 12 개의 허프만 테이블 중 하나가 선택되고, 선택된 테이블에 있는 코드 워드들과 비트열을 비교하여 일치하는 코드의 인덱스를 출력한다. 허프만 복호화 과정을 구현한 모듈의 전체 구조를 그림 2 에 나타내었다.

허프만 복호화 과정은 무손실 부호화 과정의 연산량을 덜어주어야 하므로 가능하면 적은 시간 안에 수행되어야 한다. 따라서 허프만 복호화 모듈을 MUX 로직과 게이트들의 조합으로 구성하여 전체 과정이 1 사이클 안에 수행될 수 있도록 하였다.

전체 모듈은 12 개의 허프만 테이블 모듈로 나누어

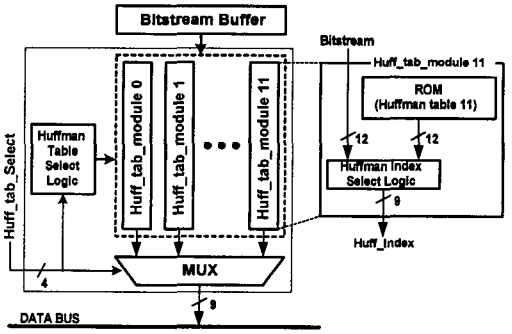


그림 2. 허프만 복호화 모듈의 구조

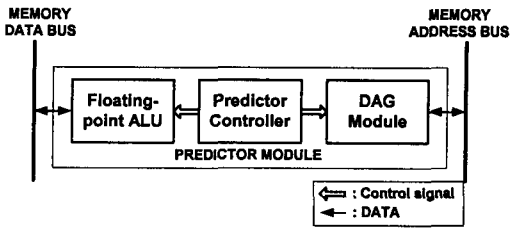


그림 3. 예측 모듈의 구조

지며 각 허프만 테이블 모듈은 허프만 테이블 ROM 과 비트열과 테이블을 비교하여 적절한 인덱스를 출력하는 선택 로직으로 구성된다.

## 3.2 예측 모듈

예측 모듈은 각 스펙트럼 성분 당 하나의 2 차 후방 예측 격자 구조를 갖는다. 예측 과정은 연산의 특성상 비교적 규칙적인 연산을 반복하는 연산 중심 과정으로서 대부분의 연산이 곱셈과 MAC 연산, 그리고 나눗셈 연산으로 구성되어있고, 이로 인해 연산의 결과값들이 넓은 동적 영역을 갖고 높은 해상도를 요구한다. 따라서 예측 과정은 고정 소수점 연산보다는 부동 소수점 연산을 사용하는 것이 유리하다. 또한 예측기가 후방 예측 구조이므로 비트열로부터 가져오는 부가 정보가 적고, 각 스펙트럼 성분 당 하나의 예측기가 존재하므로 요구되는 메모리의 용량도 막대하다. 따라서 이러한 예측기의 특성상 예측 과정을 하나의 모듈로 분리하여 다른 복호화 과정과 병렬적으로 동작시키고, 고정 소수점 연산을 사용하는 프로세서 코어와는 별도로 예측 과정을 부동 소수점 연산을 사용하여 구현하였다.

예측 과정의 알고리듬을 구현한 예측 모듈의 전체 시스템 구조를 간략히 나타내면 그림 3 과 같다.

예측 모듈은 그림과 같이 크게 부동 소수점 ALU 모듈, DAG 모듈 그리고 제어기의 3 개의 모듈로 구성된다. ALU 모듈은 실제로 예측 과정의 주요 연산을 맡고, DAG 모듈은 연산에 필요한 데이터의 주소를 발생시키는 역할을 하며, 가운데 위치한 제어기는 양쪽에 위치한 각 모듈에 제어 신호를 발생시켜서 예측 모듈의 전체 시스템 동작을 제어하는 역할을 한다.

# IV. 실험 및 결과

본 논문에서는 MPEG-2 AAC 오디오 복호화기에 대한 검증으로서 C 언어로 구현된 시뮬레이터를 사용하여 시스템의 실기간 구현 검증과 복호화 알고리듬 검증을 수행하였다.

논문에서 사용된 프로세서 코어는 최대 27MHz 의 시스템 클럭에 대해 검증이 되었고[3][4], 구현된 복호

화기는 최대 48kHz 까지의 표본화 주파수를 지원하도록 하였다. 프로세서 코어의 클럭 주파수를 $f_{CLK}$ 라고 하고 최대 표본화 주파수를 $f_S$ 라고 할 때, 하나의 오디오 샘플에 대응되는 클럭 수는 다음과 같다.

$$f_{CLK} \times \frac{1}{f_S} = 27 \times 10^6 \times \frac{1}{48 \times 10^3} = 562.5 \text{ (cycles)} \quad \cdots (4)$$

한 오디오 프레임의 샘플 수는 1024 샘플이므로 복호화 과정을 실시간으로 처리하기 위해서는 한 프레임에 대한 복호화 과정이 $1024 \times 562.5 = 576{,}000$ 사이클 안에 수행되어야 한다.

구현된 전체 복호화 시스템의 연산량을 표 1에 나타내었다. 표에서 허프만 복호화 과정은 실제 구현된 것과 같이 1 사이클에 동작하는 것으로 계산하였고 예측 과정은 프로세서 코어가 예측 모듈을 구동하는데 필요한 연산량을 나타낸 것이다. 최악의 경우에도 실시간으로 동작해야 하므로 가장 많은 연산량을 갖는 경우를 고려하여 1024 샘플의 긴 블록을 갖는 프레임에 대해 연산량을 구하였다.

예측 과정의 모의 실험 결과, 예측 모듈의 수행 시간이 장변환 프레임의 경우 약 135,000 사이클, 단변환 프레임의 경우 약 30,000 사이클 정도가 필요한 것으로 나타났으므로, 프로세서 코어 모듈이 TNS 와 필터 뱅크 과정을 수행할 동안 예측 모듈은 다음 프레임에 사용될 모든 예측 상태 변수의 값을 충분히 계산할 수 있다. 따라서 구현된 시스템은 실시간으로 MPEG-2 AAC 오디오 복호화 과정을 수행할 수 있다.

그리고 본 논문에서는 다음과 같은 두단계를 거쳐 시스템의 알고리듬에 대한 검증을 수행하였다. 시스템 설계에 앞서 C 언어로 부동 소수점 및 고정 소수점 모의 실험을 하여 복호화 알고리듬에 대한 기능적 검증을 하였다. 그리고 이러한 검증을 바탕으로 복호화 시스템을 설계하고, 구현된 시스템이 모의 실험과 같은 결과를 출력하는지 검증하였다.

그림 4는 한 오디오 프레임에 대한 시스템의 출력과 부동 소수점 모의 실험 결과의 16 비트 PCM 을 비교한 것이다. 그림에서 나타난 것처럼 두 결과는 ±2 정도의 근소한 차이 밖에 나지 않으므로 두 결과는 거의 일치한다고 볼 수 있다. 따라서 설계된 MPEG-2 AAC 복호화 시스템은 정상적으로 복호화 과정을 수행한다.

**표 1. 복호화 과정별 연산량**

| 복호화 과정 | 연산량 (cycles) | 비율 (%) |
|---|---|---|
| 무손실 부호화 | 232,211 | 42.4 |
| 채널간 부호화 | 6,720 | 1.2 |
| 예측 | 3,738 | 0.7 |
| TNS | 104,933 | 19.1 |
| 필터 뱅크 | 200,185 | 36.5 |
| 기타 | 641 | 0.1 |
| 합계 | 548,428 | 100 |

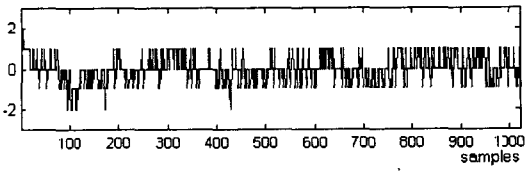


**그림 4. 시스템 출력과 모의 실험의 결과 비교**

## V. 결론

본 논문에서는 MPEG-2 AAC 오디오 복호화 시스템을 설계하였다. 구현된 복호화기는 2 채널의 메인 프로필 MPEG-2 AAC 비트열을 실시간으로 복호화하고, 32, 44.1, 48 kHz 의 표본화 주파수를 지원하며, 표준안에서 제안하는 기법 중 커플링 채널을 제외한 모든 기법을 지원한다.

설계에 들어가기 앞서 알고리듬 최적화 과정을 통해 시스템의 연산량 부담을 줄였으며, 전체 시스템을 프로세서 코어 모듈, 허프만 복호화 모듈, 그리고 예측 모듈의 3 개의 모듈로 나누어 효율적인 시스템이 되도록 구현하였다. 오디오 부호화/복호화에 적합하도록 설계된 프로그래머블 프로세서 코어를 사용하여 예측 과정을 제외한 모든 복호화 과정을 소프트웨어로 구현하였고, 많은 연산량과 자원을 요구하는 예측기와 허프만 복호화기는 코어 외부의 독립된 모듈로 구성하고 효과적인 구조를 제안하였다. 프로세서 코어는 이미 다른 부호화/복호화 시스템의 설계를 통하여 물리적으로 검증되었고, 3 개의 모듈로 구성된 전체 복호화 시스템에 대해 C 언어로 모델링된 시뮬레이터를 이용하여 복호화 알고리듬에 대한 검증과 실시간 동작에 대한 검증을 수행하였다.

이렇게 단일 칩으로 구성된 복호화 시스템은 디지탈 오디오 방식의 저장매체나 디지탈 오디오 방송 장비 등의 다양한 응용 분야에 사용될 수 있으며, 향후 MPEG-4 시스템을 구현할 때 MPEG-2 AAC 복호화 모듈로써 재사용될 수 있다.

## 참고 문헌

[1] ISO/IEC JTC1/SC29/WG11 No.1650 "IS 13818-7 (MPEG-2 Advanced Audio Coding, AAC)", Apr., 1997.

[2] M. Bosi and *et al.*, "ISO/IEC MPEG-2 Advanced Audio Coding", *J. Audio Eng. Soc.*, Vol. 45, No. 10, pp. 789-814, Oct., 1997.

[3] 정남훈, "VLSI 를 이용한 MPEG-2 오디오 부호화기 설계", 연세 대학교 석사 학위논문, Dec., 1996.

[4] 고우석, "AC-3 와 MPEG-2 오디오 공용 복호화기의 VLSI 설계", 연세 대학교 석사 학위논문, Dec. 1997.