Journal of Korea Multimedia Society Vol. 8, No. 7, July 2005(pp. 882-888)

부동 소수점 DSP를 이용한 MPEG-2 AAC 부호화기 구현

김 승 우†

요 약

MPEG-2 AAC는 이미 보다 진보한 차세대 기술로 표준화가 이루어졌다. AAC는 96-128kbps/stereo에서 CD 음질의 오디오 신호를 표현한다. 본 논문은 고음질의 MPEG-2 AAC LC Profile 부호화기 구현에 관하여 논하였다. 공통 스케일팩터와 무손실코딩은 각각 45%와 27%의 TMS320C30 명령어 이득을 가져왔다. 구현된 부호화기는 프로그램 메모리 7.5 kWords, 데이터 롬 18kWords, 데이터 램 92kBytes를 사용한다. 주관적 음질평가결과는 96kbps 스테레오에서 얻어진 AAC 부호화기 음질이 MP3 128kbps 스테레오에서 얻어진 것과 동일한 음질을 가짐을 보여준다.

MPEG-2 AAC Encoder Implementation Using a Floating-Point DSP

Seung-Woo Kim†

ABSTRACT

MPEG-2 Advanced Audio Coding(AAC) has already been standardized as a sophisticated next-generation technology. AAC provides an audio signal that has CD quality at 96-128kbps/stereo. This paper describes a high-quality and efficient software implementation of an MPEG-2 AAC LC Profile encoder. Common scalefactor and noisless coding are accelerated by 45% and 27%, respectively, through the use of TMS320C30 instructions. The implemented encoder uses 7.5kWords of program memory, 18kWords of data ROM and 92kBytes of data RAM, respectively. The results of subjective quality test showed that the sound quality achieved at 96kbps/stereo was equivalent to that of MP3 at 128kbps/stereo.

Key words: Audio Coding(오디오 코딩), Huffman Coding(호프만코딩), Audio Signal Processing(오디오 신호처리), AAC Encoder(AAC 부호화기)

1. 서 론

아직까지도 많은 사람들에게 알려진 오디오 압축 방법은 MPEG-1 layer Ⅲ, 즉 MP3이다. 그러나 차세대 오디오 산업에서는 1998년 11월 MPEG-4의 오디오 국제 표준의 한부분으로서 AAC을 채택하고 있으며 또한 AAC는 새로운 멀티채널 오디오 부호화 방식으로써 폭넓게 이용될 것이다. AAC는 MP3보다 30%더 작은 데이터 율에서 조차도 더 우수한 음질을 자랑한다[7].

이제까지 MPEG-2 AAC Encoder는 고정소수점 DSP칩과 인텔칩을 사용해서 구현되어왔고 부동소수점은 거의 사용하지 않았다. 현재까지 개발된 주요 기술은 MDCT 코어 마이크로프로그램 방식의 벡터 프로세서를 개발한 사례가 있다. 또한 지금까지 복호화기에서는 다양한 연구가 있어왔지만 부호화기에서는 많은 연구가 이루어지지 않았다. 그러나 AAC 부호화기에 대한 수요가 커지고, 가전제품과 컴퓨터

※ 교신저자(Corresponding Author) : 김승우, 주소 : 부산시 남구 남부산우체국 사서함21(613-600),전화 : 054)471-3440, FAX : 054)460-8609, E-mail : jejeswkim@empal.com
접수일 : 2004년 6월 24일, 완료일 : 2005년 1월 5일
† 국방품질관리소 연구원

에서 뿐만 아니라 휴대용 오디오 플레이어에서 AAC 부호화기를 내장시키려는 추세이다[13].

AAC는 MP3에 비해 크게 두지지 측면에서 장점을 가지고 있다. 첫 번째는 한 프레임 크기가 유동적이라는 것이다. 한 프레임이 압축률에 따라 비트 크기가 변하므로 전체 파일의 용량이 많이 줄어들게 된다. 실제로 MP3 파일과 비교해서 최대 30%까지 크기를 줄일수 있다[5]. 두 번째는 AAC는 양자화 오차 보정기술인 TNS와 프리딕션이라는 두가지 기법을 통해 음질을 향상한다는 점이다[2,6].

보다 우수한 품질의 오디오와 이미징 수요가 높아짐에 따라 시장은 부동소수점 DSP 칩이 제공하는 32비트 정밀도로 옮겨가고 있다. 또한 C6713는 차세대 멀티미디어 제품을 수용한 획기적인 부동소수점 성능을 제공한다. C3x는 기존의 것보다 성능은 2배 가격은 절반으로 낮아지고 있는 상황이다. 본 논문에서는 Peripheral, 내장형 메모리, 네트워크카드, 스마트카드 등을 포함하는 SOC(System On Chip) 구현을 위해 부동소수점에서 AAC 부호화기를 구현하였다. AAC 표준문서는 부호화기 절차 및 비트스트림 포맷을 제공하지만 최적화되지 않은 알고리즘을 표현하고 있다. AAC 부호화기는 루프를 돌며 주어진 비트수내에 양자화시 생기는 오차를 마스킹 하기 위해 적합한 호프만코드를 찾는 부분이 있어서 고음질을 가지면서 속도를 빠르게 하기 위해서는 알고리즘에 대한 연구가 필수적이다.

본 논문은 양자화 과정과 무손실 과정에서 알고리즘을 최적화시키는 방법과 그것을 통해 속도를 향상시키며 구현하는 방법 그리고 검증과정을 보여주었다. 과거에 효율적으로 호프만 코드를 찾으려는 노력은 많이 하였지만 비트를 줄이는 곳으로 촛점이 맞추어졌기 때문에 최적의 호프만 북을 찾는 것은 아니였다. 그러나 제안된 호프만 코드를 찾기 위한 병렬 방법은 코드북 인텍스를 한번만 계산해서 두개의 북에 대해 코딩 했을때의 결과를 얻을 수 있다. 이 방법은 보다 효율적이면서 속도를 개선시킨 사례라고 할 수 있다.

AAC는 넓은 범위의 샘플링 주파수(16Khz~96Khz)를 지원하고 있다. 또한 MPEG-2 AAC는 매우 낮은 비트율에서 방송 음질 수준의 오디오를 제공하기 위하여, 고해상도 필터뱅크와 예측기법, 허프만(Huffman) 부호화, 심리 음향 모델 등을 결합하여 사용한다[1,8].

2. MPEG-2 AAC

AAC Encoder의 전체 블럭도를 그림 1에 나타내었다. 여기에는 선택 블럭들이 많이 포함되어있고, 또한 이들은 AAC가 어떤 profile을 쓰느냐에 따라서 달라진다. MPEG-2 AAC는 Main, LC(Low Complexity), SSR(Scaleable Sampling Rate) Profiles들을 제공한다[10].

주요 블럭에 대하여 간단한 설명을 하면 다음과 같다.

2.1 비트스트림 형식

AAC 파일은 제일먼저 32비트의 인식비트를 가지고 있으며, copyright id의 존재여부와 넘버 그리고 샘플링 주파수 및 프로파일등에 대한 정보가 헤더비트에 담겨 있다.

2.2 심리음향 모델

심리 음향 모델을 이용하여 각 서브밴드에서 원음에 의해 마스킹되어 들을 수 없는 최대의 잡음 레벨을 결정할 수 있다.

최종 출력은 SMR(Signal-to-Mask Ratio)로써 나중에 iteration loop에서 bits할당을 할때 쓰여질 중요한 data를 구하게 된다.

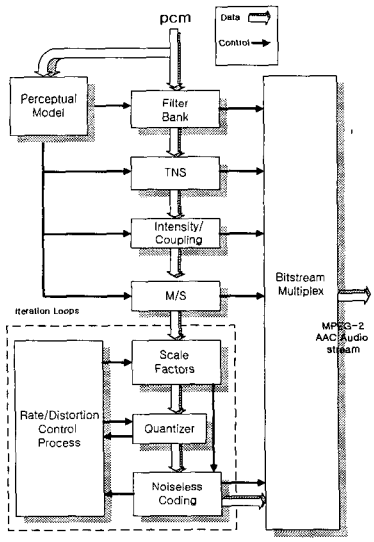

그림 1. MPEG-2 AAC 부호화기 블럭

2.3 MDCT

MPEG-2 AAC에서는 시간영역의 신호를 주파수 영역의 표현으로 바꿀 때 MDCT(Modified Discrete Cosine Transform)를 수행한다. MDCT는 시간영역에서 에일리어싱 제거(TDAC : time-domain aliasing cancellation) 기법을 사용한다[9,10].

2.4 TNS

TNS개념은 MPEG-2 AAC에서 새로이 도입된 개념이다. MP3나 MPEG-2 BC와 같은 오디오 부호화 기술로는 과도 구간의 신호와 피치가 많은 신호에 대해서 해결되어야 할 문제가 있었다. 이 문제는 마스킹 임계값과 양자화 잡음 사이의 시간적 불일치로 인한 프리 에코(pre-echo) 현상 때문에 지각적 부호화가 어렵기 때문에 발생하는 것이다.

TNS 기술은 이러한 문제를 해결하기 위해 필터뱅크 윈도우 내에 있는 양자화 잡음의 시간적 미세 구조를 제어한다.

2.5 Iteration loop

이 루프의 기본은 analysis by synthesis이다. 그렇기 때문에 많은 연산량을 필요로 하는 부분이다. 그러나 loop 횟수가 많아질수록 항상 최적의 상태로 수렴하지는 않는다. common_scalefactor와 scalefactor 이 두가지 value에 따라 양자화 스텝사이즈가 결정되며, 양자화시 distortion이 어떤 모양으로 발생하느냐에 따라서 masking curve아래로 masking되느냐 그렇지 못하느냐에 대한 문제까지 달려 있다. 그리고 또한 이 factor의 상태로 양자화를 하여 coding 했을 때 우리가 사용할 수 있는 bits 범위 내에서 bit cost를 유지할 수 있는지가 관건이 된다.

3. 하드웨어 구현을 위한 알고리즘 최적화

3.1 Non-linear Quantization 과정의 최적화

MPEG-2 AAC 비선형 양자화기 식은 다음과 같다.

```
x_quant=int((abs(mdct_line)*(2^(1/4*(sf_decod
er-SF_OFFSET))))^(3/4)+MAGIC_NUMBER)
```

(1)

식(1)을 다음과 같이 나타내면

$$pow\_quant[i] = |i|^{\frac{3}{4}} \quad (2)$$

i의 범위는 $0 \leq |i| \leq 8191$와 같다.

표 1. 양자화 과정에서 입력 값 범위

구분	양자화 과정시 입력이 127 이내에 들어오는 회수(총: 4096)	비율(%)
Ballade	4096	100
Classic	4096	100
Pop	4087	99.78
Toeic	4003	97.7
OST	4096	100

대부분의 입력이 0에서 127에 집중되어있다. 따라서 양자화 과정에서 들어오는 입력 값이 0에서 127까지는 테이블로 만들고 나머지는 Linear Interpolation을 사용하여 구현하였다.

3.2 양자화시 스케일 펙터 대역 최적화

각 스케일 펙터 대역내의 왜곡을 계산하기 위해서 곱해주는 성분(Factor)는 다음 식과 같다.

$$2^{-1/4 * (scalefactor(sb) - common\_scalefac)} \quad (3)$$

여기에서 2의 지수 승으로 있는 성분 중에 scalefactor(sb) - common_scalefac의 범위는 표본화 주파수에 따라 달라지지만, 0에서 -100사이의 정수값이다.

따라서, $2^{-1/4*(0)}$ 에서 $2^{-1/4*(-100)}$사이의 값만 테이블로 만들면 간단히 해결할 수 있다. 따라서, 100개의 table로 해결할 수 있다.

3.3 Iteration 과정의 최적화

Iteration loop는 가장 많은 load를 차지하는 부분으로 inner Iteration loop와 outer iteration loop가 함께 맞물려 돌아가면서 최적의 양자화 스텝사이즈를 찾는 곳이다. 내부 반복 루프의 탈출조건은 무손실부호화에서 사용된 비트수가 사용가능한 비트수보다 작을 경우이고, 외부 반복 루프의 탈출조건은 그림 2에 나타내었다.

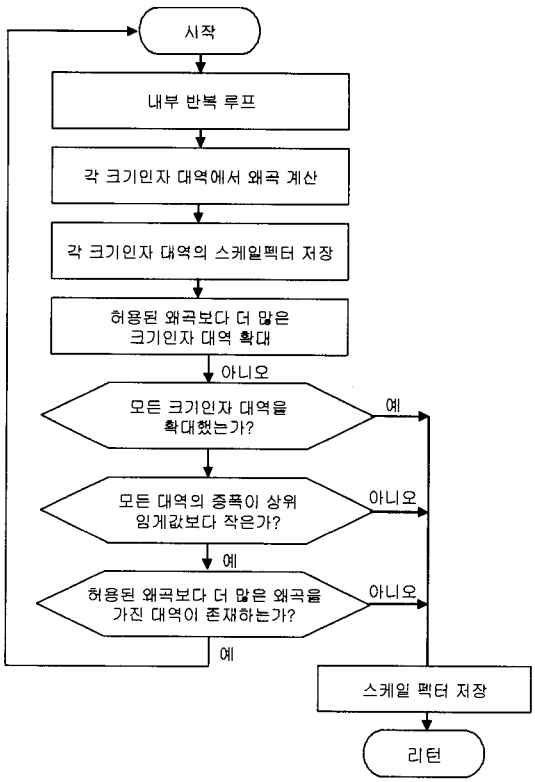

그림 2. MPEG-2 AAC 외부 반복 루프

3.3.1. Common scalefactor와 scalefactor값의 조정

외부반복 루프에서 초기에 어떤 scale factor 밴드도 증폭되지 않는다. 내부반복루프가 호출되고 각 scale factor 밴드에 대해 양자화시 야기된 왜곡이 SMR과 비교된다. 그 결과가 현재 루프까지 최고라면 scale factor가 저장된다. 허락된 왜곡보다 더 많은 왜곡을 가진 밴드는 증폭된다. 만약 모든 scale factor 밴드가 마스킹 레벨 아래로 들어온다고 해도 허락된 bit보다 더 많은 bit를 요구한다면 다시 내부 루프에서 global gain을 변화시켜야 한다.

Inner loop를 처음 들어가면 최소한 3번의 common scalefactor값이 변화한다. 이 값이 너무크면 루프수는 작게 돌지만 잡음이 많고, 너무 작으면 잡음은 작지만 루프수가 많게 된다. Ballade, Classic, Pop, Toeic, OST 5종을 일정프레임 입력으로하여 초기 global gain을 변경시키며 루프회수를 측정하고, 이때 통계적으로 common scalefactor의 값을 평균을 내보니 표 2와 같았고 64로 했을때 loop를 수행하는 횟수가 가장 작고 잡음도 작았다. 위의 샘플 데이터 중 프레임당 가장많은 계산을 하는 것을 기준으로 연산량 감소를 계산하였다.

표 2. 초기양자화 스텝사이즈와 잡음 및 루프횟수

start gobal gain	the number of inner loop	the number of outer loop	total noise
공식	4919	725	47729.804
이전frame의 값사용	1981	695	49151.014
10	10578	725	47729.804
20	8618	725	47729.804
30	6658	725	47729.804
40	4698	725	47729.804
50	3047	725	47729.804
60	1961	732	47541.89
64	2316	739	47336.181
70	606	196	68156.447
80	588	196	68156.447

또한 scalefactor을 현재 프레임의 마스킹에 가장 근사한 값으로 만들고 초기 loop를 들어가게 되면 가장 조금만 loop를 돌고도 distortion을 마스킹하고 bits를 적절히 할당할 수 있었다.

표 3. common scalefactor의 변경 전과 후의 비교

구분	연산량(Cycles)	MIPS
start global gain 변경전	263,177,448	113.34
start global gain 변경후	143,965,076	61

3.3.2 Modified noiseless coding 방법

noiseless coding block에서는 초기에 현재 spectral section에서 어떤 호프만 코드 북을 이용할 수 있는지 보기 위하여 그 section내의 coefficient들 중 가장 큰 절대 값을 가지는 것을 찾는다. 그리고, 각 spectral section에 대한 dynamic range가 계산된다. 여기서 한 section은 그것의 spectral data의 dynamic range보다 같거나 더 큰 dynamic range를 가지는 호프만 북을 가지고 부호화 한다. 그러나 dynamic range가 16보다 크다면 11번째 ESC codebook을 선택한다. 그 값에 따라서 알맞은 코드 북 n과 n+1번째 호프만 코드 북을 사용했을 때의 필

요한 bit수를 계산 한다. 그리고 그 값을 버퍼에 저장한다. 이때 사용되는 함수는 output_bits함수이다. 두개의 code book(n,n+1)을 사용 했을 때 더 작은 bit을 필요로 하는 book 번호와 total bit cost를 한 scalefactor band마다 구해서 리턴하는 과정이 필요하다. 그런데 이 과정이 inner iteration loop의 하나이므로 많은 계산량을 필요로 한다. 따라서, 구현을 위해서는 이부분의 계산량을 줄이는 것이 필수적이고 아래의 그림 3에 그 과정들이 제안되었다.

그런데, 사실 이 과정은 계산만 할뿐 실제 writing buffer에 inner iteration loop가 어떤 특정한 값을 써넣지는 않는다. 임의의 한 섹션에서 Maximum absolute value 의 값이 같은 코드 북을 사용하는 것들은 호프만 북에서 length와 코드워드를 구하기 위해서 index를 계산할 때 같은 index를 사용한다는 것을 알수 있다. 따라서 만약에 n 번째 book을 가지고 coding 했을 때와 n+1번째 book을 가지고 coding 했을 때 두 번다 같은 index를 사용하는데 이 index를 중복해서 계산한다는 것은 계산량에 있어서 낭비가 되고, output_bits함수를 writ_flag가 1일때와 0일때를 나누지 않고 같이 사용한다는 것은 효율적이지가 못하다. 코드 북 0는 그 섹션의 모든 coefficients가 0임을 나타낸다. 코드 북 11은 양자화된 계수의 절대값이 16보다 큰 것을 표현하기 위해 사용된다. 이렇게 코드북 인덱스가 0 또는 11이 아닌 그사이에 쓰이는 것들에 대해서는 한번의 output_bits블럭에서 두개의 북에 대해 coding 했을때의 결과를 얻을수 있도록 최적화를 시켰다.

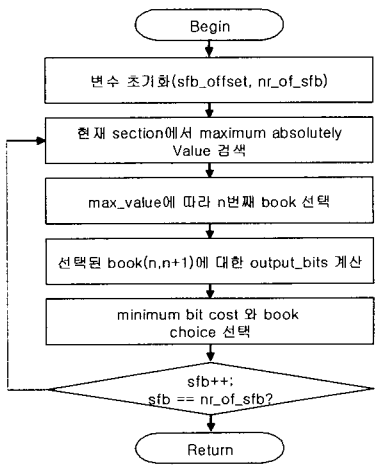

그림 3. 제안된 Noiseless 코딩 블럭

표 4. noiseless coding 블럭의 변경 전과 후의 비교

구분	연산량(Cycles)	MIPS
noiseless coding 변경전	143,965,076	62
noiseless coding 변경후	104,490,781	45

즉 코드 북 인덱스를 한번만 계산해서 두개의 북을 코딩했을 때의 coding 결과를 출력 하므로써 계산량은 20~25%가량 줄일수 있었다.

4. MPEG-2 AAC DSP 구현과 검증

4.1 Board상의 구현

AAC 부호화기를 위하여 제조된 TMS320C30은 상용의 4단이 아닌 6단 파이프라인을 가지고 있고, 114종의 명령어를 가진다. 어드레스 버스 24bit와 데이터 버스 32bit를 가진다. 3연산 어드레싱 모드외 7개의 어드레싱 모드를 가지며, 다양한 레지스터와 메모리를 가지고 있다. DSP로 구현되는 어셈블리코딩을 위해서 TMS320C3x의 Code composer tool을 사용하였다. 그림 4에 알고리즘 구현 과정중 사용한 Code composer를 보였다. 우선 Frame 이 부호화 될 수있는 데이터 엔코딩 블록을 코딩한 후 메모리 초기화 과정을 삽입한다. 또한 Serial Port initialization을 추가한다. 가장 중요한 것 중 하나가 circular input buffer를 만들고, 그리고 RBUF_RCOUNT와 RBUF_WCOUNT를 만들어서 input buffer의 사용량과 비어있는 량을 항상 체크 하게 만든다. 원래는 wav file을 분석하여 몇 프레임을 계산해야 임의의 한곡에

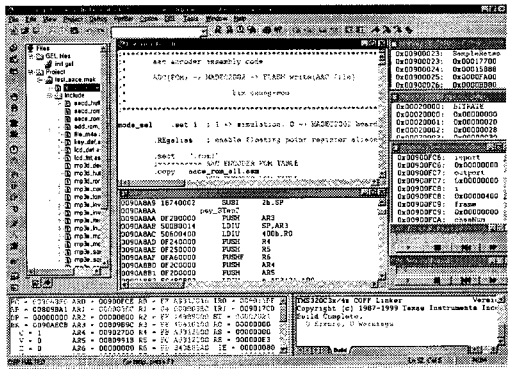

그림 4. Code Composer 어셈블리 컴파일 환경

대한 부호화 과정을 마칠 수 있도록 되어 있었던 것을 raw data 입력으로 무한 루프를 돌면서 입력 버퍼가 한 프레임보다 더 많은 양을 쓰고 있으면 한 프레임을 받아들이도록 만들었다. 정해진 input buffer의 양은 없지만 buffer 의 오버플로우가 생기지 않는 범위로 프레임 크기의 두배를 잡았다. 이렇게 해서 인터럽트와 호환하면서 AAC 프로그램 코드가 돌게 되어 있다[3,4].

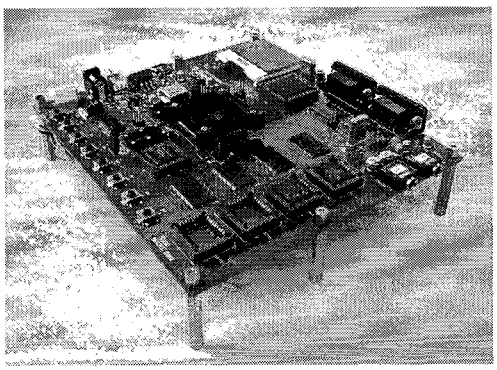

그림 5. Board 구성도

4.2 주관적 음질평가

Simulator와 칩에서의 결과는 Bit-exact하게 일치함을 볼수 있었다. Board상에서 encoding이 끝나고 PC 로 upload된 결과는 AAC표준 codec을 넣고 주관적 청음 시험을 하여 검증하였다. 청음테스트에 참여한 사람은 음향분야에서 경험이 있는 9명으로 구성했고, CMOS(Comparison Mean Opinion Score)시험 방법을 사용했다[11]. 청취자는 매번 3가지의 테스트 데이타 즉 Ref, AAC, MP3를 들었다. Ref는 원음이고, AAC 와 MP3는 Coding과정을 거친 것이였다. 청취자는 AAC 와 MP3가 들려지는 순서를 알지 못하도록 하였고, 7단계 비교 등급을 사용하여 청음결과를 표시하도록 하였다. 청음시험은 외부와의 소리가 단절된 곳에서 헤드폰을 착용하고 시험하였다. 시험에 사용된 MP3 엔코더는 고음질을 가지는 것으로 알려진 상용 소프트웨어 제품을 사용하였다. 테스트 샘플 데이터는 Castanets(A), Pop Music(B), Glockenspiel(C), Pitch Pipe(D), Suzanne Vega(E)의 5종을 사용했다. 테스트 결과 그래프에서 보여지는 숫자와 같이 AAC 부호화기 음질은 똑같은 96kpbs에서 시험하였을때 MP3 보다 훨씬 좋다는 것을 확인할수 있었고, MP3 128kbps와 비교했을 때는 조금더 좋다는 것을 확인할 수 있었다.

표 5. 등급 비교 점수표

AAC is much better than MP3	+3
AAC is better than MP3	+2
AAC is slightly better than MP3	+1
AAC is the same as MP3	0
AAC is slightly worse than MP3	-1
AAC is worse than MP3	-2
AAC is much worse than MP3	-3

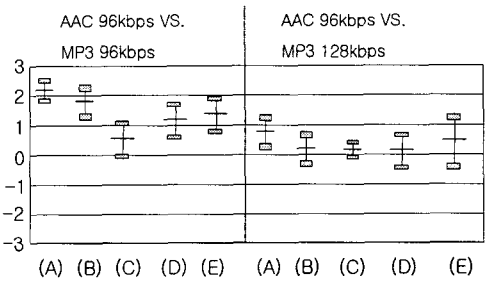

그림 6. 주관적 음질평가 결과

5. 결 론

본 논문에서는 차세대 오디오 압축 포맷으로 다가올 MPEG-2 AAC 부호화기의 TMS320C30 구현에 관한 내용을 담고 있다. 이 과정에서 가장 많은 계산량을 차지하고 있는 iteration loop의 최적화 과정을 위해 제안된 몇가지 방법을 소개 하였다. 주관적 음질 평가를 수행하였고, 그 결과 MP3 보다 우수한 음질을 얻을 수 있었다. C 언어로 작성된 AAC encoder를 최적화후 어셈블리 언어로 구현을 하였다. 그것을 상용 DSP보다 성능이 우수하게 설계된 TMS320C30

표 6. MPEG-2 AAC 부호화 과정 시 프레임당 걸리는 사이클 수

구분	Cycle 수	Percentage(%)	MIPS
심리음향모델	278,616	16	12
필터뱅크	208,962	12	9
TNS	139,308	8	6
Iteration Loop	998,374	57	44
기타	116,090	7	5
Total	1,741,350	100	76

표 7. 사용된 메모리 양

메모리종류	Size
Program ROM	7.5kWords
Table ROM	18kWords
RAM	92kbyte

을 장착한 Board상에서 구현하였다. AAC 부호화기의 계산량은 76MIPS정도가 나왔으며, 사용되는 데이터 ROM은 18Kwords, 데이터 RAM은 92kByte 였으며, 프로그램 코드는 7.5Kwords를 사용했다.

향후 MPEG-2 AAC는 디지털 오디오 장비나 디지털 오디오 저장매체 등의 다양한 응용분야에서 사용될 수 있으며, 또한 AAC player 등의 제품을 탄생시켜 새로운 시장을 개척해 나갈 수 있을 것으로 생각된다. 향후의 연구방향은 SOC(System On Chip)을 구현하는 일이다. 오디오 부호화 시스템을 SOC로 구현한다면, 디지털 오디오 장비나 디지털 오디오 저장매체 등의 다양한 응용분야에서 사용될 수 있을 것으로 예상된다.

참 고 문 헌

[1] ISO/IEC 13818-7, "Generic Coding of Moving Pictures and Associated Audio, Part 7 : Advanced Audio Coding(AAC)," Mar. 1997.

[2] Ivan Dimkovic, "Fast implementation of AAC LC Encoder," *PSYTEL RESEARCH* 2001.

[3] TEXAS INSTRUMENTS "TMS320C3X User's Guide," July 1992.

[4] TEXAS INSTRUMENTS "TMS320C3x/C4x Assembly Language Tools," Digital Signal Processing Solutions.

[5] ISO/IEC 11172-3, "Coding of Moving Picture and Associated Audio for Digital Storage Media at up to about 1.5Mbit/s-CD Part3. Audio," Aug. 1993.

[6] Karlheinz Brandenburg, "MP3 AND AAC EXPLAINED," *AES 17th International Conference* on *High Quality Audio Coding.*

[7] "MPEG Digital Audio Coding," *IEEE SIGNAL PROCESSING MAGAZINE*, Semptember 1997.

[8] M. Bosi, K. Brandenburg, S. Quackenbush, L. Fielder, k. Akagiri, H. Fuchs, M. Dietz, J. Herre, G.A.Davidson, and Y. Oikawa, "ISO/IEC MPEG-2 Advanced Audio Coding," *Audio Eng. Soc.* Vol. 45. No. 10. October. 1997.

[9] Vladimir Britanak and K. R. Rao, "An Efficient Implementation of the Forward and Inverse MDCT in MPEG Audio Coding," *IEEE SIGNAL PROCESSING LETTERS*, VOL. **8**, No.2, February 2001.

[10] Byeong Gi Lee, "A New Algorithm to Compute the Discrete Cosine Transform", *IEEE Trans. on Acoust, Speech and Signal Processing*, Vol. ASSP 32, No. 6, pp. 1143-1245, 1984.

[11] B. Edler, J.Herre, and K. Brandenburg, "Core experiment methodology for MPEG-4 audio," *ISO/IEC JTC1/SC29/WG11*, N1748, Jul. 1997.

[12] J.D.Johnston, "Transform Coding of Audio Signals Using Perceptual Noise Criteria," *IEEE J. on Selected Areas in Communication*, 6,2, Feb. 1998.

[13] T.Nomura and Y.Takamizawa, "Processor-Efficient Implementation of a High Quality MPEG-2 AAC Encoder," 110th *Audio Engineering Society Convention Paper*, May 2001.

김 승 우

2001년 2월 금오공과대학교 전자공학과(공학사)

2003년 2월 부산대학교 전자공학과(공학석사)

2003년 ~ 현재 국방품질관리소 연구원

관심분야 : 디지털 오디오 신호 처리, DSP 설계, 소프트웨어 품질보증