고정소수점 DSP(ADSP-2181)을 이용한 실시간 G.723.1 음성부호화기 개발에 관한 연구

A Study on the Development of the Real-Time G.723.1 Speech Codec Using a Fixed-Point DSP(ADSP-2181)

박정재 · 정익주*

(Jung-jae Park · Ik-joo Chung)

ABSTRACT

This paper describes the procedure of implementing a real-time speech codec, G.723.1 which was developed by DSP Group and standardized by ITU-T, using fixed-point DSP, ADSP-2181. This codec has two bit rates associated with it, 5.3 and 6.3 kbit/s. We implemented only one bit rate, 6.3 kbit/s, of the two with fixed-point 32-bit precision. According to the result of the experiment, the amount of computational burden is about 55 MIPS and its quality is similar to the result of the PC simulation with floating-point arithmetic. In this paper, we proposed a method to use a fixed-point DSP and a procedure for developing a real-time speech codec using DSPs and finally developed a G.723.1 speech codec for ADSP-2181.

Keywords: speech coding, G.723.1, fixed-point DSP

I. 서 론

음성부호화기는 응용 분야에 따라 전송률, 연산량, 오류에 대한 음질저하, 지연 등을 고려하여 가능한 최고의 음질을 유지하도록 설계되어 사용된다. 본 논문의 G.723.1 부호화기는 toll 정도의 음질로 가능한 한 낮은 전송률을 갖도록 설계되었다. 주된 목적은 음성신호를 주로 하는 멀티미디어 신호를 낮은 전송률을 갖는 채널을 통하여 전송하는 것인데, 예로 LAN을 통한 화상전화기(H.323)나 기존 전화망을 통한 화상회의 시스템(H.324) 및 인터넷 폰 등에 사용된다.

이 부호화기는 5.3과 6.3 kbps의 두 전송률을 갖는데 부호화기와 복호화기에서 이 둘을 모두 지원하도록 표준으로 정해졌다. 음질은 MOS 시험결과 3.98로 고음질이며, 기본지연은 37.5 ms로 큰 편이다. 구현에 필요한 연산량은 일반적인 합성에 의한 분석(Analysis-by-Synthesis) 방식과 마찬가지로 크다. 단일 정밀도로 구현하는 경우 연산량 20 MIPS, ROM 16 kword, RAM 2~3 kword 가량이 필요하다.

이 논문에서는 연산기로 Analog Devices사의 ADSP-2181을 사용했는데, 이 연산기는 33

* 강원대학교 공과대학 전자공학과

MIPS의 연산속도와 프로그램 메모리 16 kword, 데이터 메모리 16 kword를 내장하고 있다. 따라서 G.723.1 부호화기를 실시간으로 처리할 수 있는 성능을 가지고 있다. 더욱이 이 연산기는 효율적인 구조를 가지고 있어 어느 정도 연산량의 이득을 볼 수 있기 때문에 정밀도를 높여 연산을 함으로써 구현 음질을 높일 수 있다. 16비트 연산만으로 구현 한 경우, 유한 단어길이 효과(finite wordlength effect)에 의한 잡음이 현저하게 나타나게 되어 음질에 막대한 영향을 미치게 되므로, 가능한 한 연산정밀도를 양자화 잡음 범위 내로 줄이는 것이 필요하다. 이러한 문제는 부호화기 구조상의 문제가 아니라 구현상의 문제이기 때문에 구현 방법에 따라 음질의 차이가 크게 발생할 수 있다. 이 논문의 부호화기를 제작하는 과정에서는 제한된 성능의 연산기로 구현하는데 발생하는 음질의 저하를 줄이기 위해 여러 조건들을 고려하였다.

구현을 위해 먼저 PC상에서 시뮬레이션을 하고, 이것을 바탕으로 하여 DSP로 구현한 후 두 결과를 비교하였다. DSP에서 32비트로 구현한 결과는 PC상에서 단정도 실수로 구현한 결과와 음질은 비슷했으나, 55 MIPS가량의 연산량이 소요되었다. ADSP-2181은 33 MIPS의 성능을 가지고 있기 때문에 상당한 연산량 감소 대책이 필요했는데, 먼저 음질을 고려하여 정밀도 감소나 최적화 시킬 알고리즘을 선택하였다. 각 알고리즘은 양자화 잡음을 갖게 되는데, 정밀도에 대한 그 정도가 다르다. 따라서 연산량을 기준으로 음질에 미치는 민감도, 즉 양자화 잡음과의 관계를 고려하여 정밀도를 감소시키고 구현 알고리즘을 최적화 하였다.

이 논문은 다음과 같이 구성된다. 2장에서는 G.723.1의 개요에 대하여, 3장에서는 구현 방법에 대하여, 그리고 4장에서는 구현 결과에 대하여 다룬다. 마지막으로 5장에서는 결론을 맺는다.

II. G.723.1의 개요

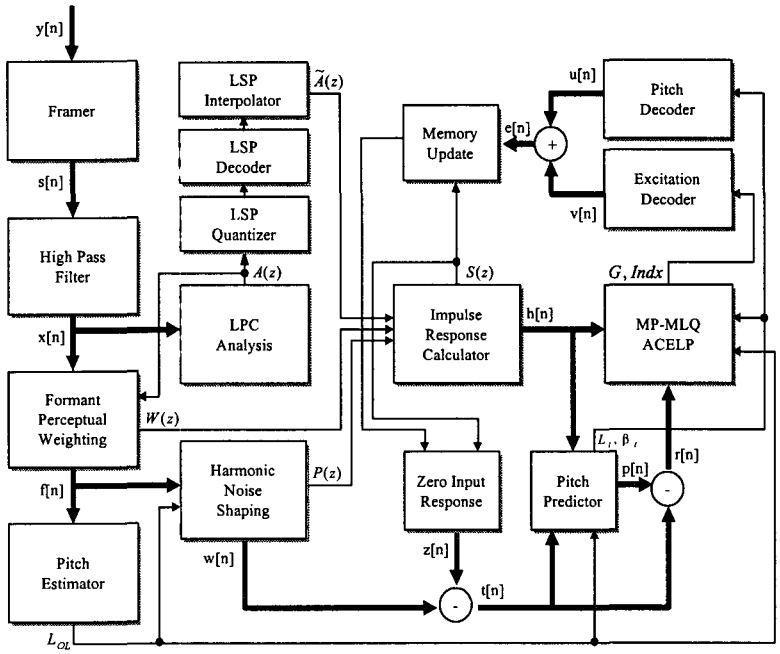
G.723.1은 DSPGroup의 TrueSpeech 를 기반으로 1995년에 ITU-T에 의해 표준화되었다. 낮은 전송률의 채널을 통하여 음성정보를 주로 하는 멀티미디어 신호를 교환하기 위해 개발되었는데, 낮은 전송률에서 고음질을 유지하기 위해 다양한 기법이 사용되었다. 혼성 부호화기 계통으로, 기본구조는 분석에 의한 합성 방법을 사용한다. 다양한 알고리즘과 테이블의 사용으로 복잡하며, 낮은 전송률을 유지하기 위해 30 ms의 긴 분석 프레임을 사용하여 지연이 크다. 사용된 알고리즘은 5.3 kbps의 경우 ACELP(Algebraic Codebook Excited Linear Predicrion)방식을, 6.3 kbps의 경우 MP-MLQ(Multi-Pulse Maximum Likelihood Quantization)방식을 사용한다. 가변 전송률로 경우에 따라서 적당히 두 전송률 중 하나로 부호화를 한다. G.723.1 부호화기의 특징을 요약하면 표 1과 같다.

부호화기의 구조는 그림 1과 같고, 복호화기는 그림 2와 같다. 본 부호화기에서 사용되는 신호는 8 KHz, 16 bit으로 샘플링된 신호이다. 입력된 신호의 샘플은 240샘플 단위로 분석되며, 이것을 다시 60 샘플의 4개 서브 프레임으로 나누어 분석한다. 분석 과정은 그림 1의 부호화기 구조에 보인 것과 같다.

표 1. G.723.1의 특징

전송률	5.3 / 6.3 kbps
음질	3.98 (MOS)
복잡도	20 MIPS, ROM 16 kword, RAM 16 kword
지연	37.5 ms (frame: 30 ms, look ahead: 7.5 ms)
알고리즘	ACELP(5.3 kbps), MP-MLQ(6.3 kbps)

그림 1. G.723.1음성부호화기의 구조



먼저 입력 샘플을 High-Pass Filter를 통과시키고, 10차 LPC분석에 들어간다. 분석은 서브프레임 단위로 실시하며, 180샘플 Hamming Window를 사용한다. 마지막 서브프레임에 대한 파라메터만을 LSP 양자화하여 전송한다. LSP 양자화에는 PSVQ(Predictive Split Vector Quantization) 방식을 사용한다. 복호화에서는 각 서브프레임 단위로 LSP계수를 선형 보간하여 사용한다. 여기서 LPC 필터의 안정성은 음질에 상당한 영향을 미치므로 LPC계수를 구하는 과정과 LSP 양자화를 하는 동안 강인성을 갖도록 부가처리를 하거나, LSP를 역양자화하고 선형보간하는 동안 필터의 안정성을 검사하여 안정성을 갖도록 보정한다.

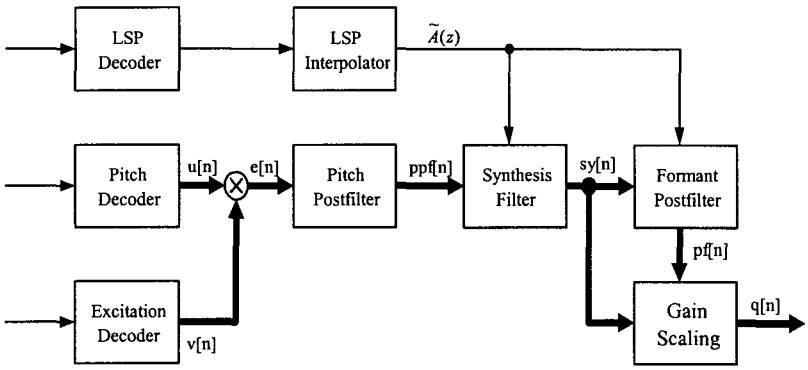
LPC를 예측한 후 여기신호의 부호화 과정에서는 분석에 의한 합성(Analysis-by- Synthesis) 방식을 이용한다. 이 과정은 많은 연산량을 요구하며 음질에 커다란 영향을 미친다. 합성에 의한 분석에서 합성음과 원음 사이의 주관적 음질 비교를 위해 지각가중필터를 적용하여 noise shaping을 실시한다. 이 필터는 formant와 harmonics 모두를 감쇄시킨다.

주기성을 모델링하기 위해 5차 피치 예측기를 사용하는데, 먼저 open-loop로 근접 피치구간을 찾고 그 주변에서 가장 최적의 예측이 가능한 필터예측계수를 찾는다. 즉, 피치구간과 필터계수인 이득값을 변화시키면서 예측오차가 최소인 계수를 선택한다. 필터계수는 벡터양자화하기 때문에 표준 테이블이 주어져 있으며, 5차 170(85)개의 벡터로 구성되어 있다.

피치예측 후 잔류신호의 부호화에는 5.3 kbps 모드에서 ACELP방식, 6.3 kbps 모드에서 MP-MLQ방식을 이용한다. 두 방식 모두 잔류신호를 몇 개의 펄스로 근사화 하는데, 펄스는 모두 같은 절대값(혹은 이득값)을 갖고 부호만 다르다. 이 펄스의 위치와 부호 및 이득값을 구하기 위해 합성을 통해 선택한다. 모든 경우에 대하여 모두 찾는 경우 너무 방대한 연산량이 요구되므로 적당한 제한을 두어 찾는다. MP-MLQ방식에서는 위치가 모두 홀수이거나 모두 짝수라고 가정하며, ACELP 방식에서는 코드북(실제 테이블로 존재하는 것이 아니라 펄스 위치를 제한)에서

찾는다. 연산량은 ACELP에 비해 그 찾는 제한 조건이 적은 MP-MLQ 방식이 더 많다. 음질은 MP-MLQ 방식이 더 뛰어난데 그 이유는 사용 펄스 수가 더 많고(ACELP는 4개, MP-MLQ는 5~6개), 펄스를 찾는 제약이 적기 때문이다.

그림 2. G.723.1 복호화기의 구조



복호화기는 그림 2와 같다. 전송된 정보로 여기신호를 구성하고, 합성필터를 거치면 합성음이 만들어진다. 지각가중필터에서 noise shaping한 것에 대응하여 postfiltering을 실시하여 formant와 harmonics를 향상시킨다.

III. ADSP-2181을 이용한 구현

3.1 ADSP-2181의 특징

ADSP-2181은 33 MIPS의 연산속도와 16 kword의 프로그램 메모리 및 16 kword의 데이터 메모리를 가지고 있다. 기본적으로 한 연산은 한 클럭에 수행되며, 경우에 따라서는 3개의 연산을 한 클럭에 수행할 수도 있다. 연산기의 구조는 다양한 기능을 제공하면서도 단순하고 명료하며, 어셈블리 니모닉 또한 산술 수식과 유사하여 사용법을 쉽게 익힐 수 있다.

연산기로는 곱셈을 제외한 사칙연산과 논리연산을 행하는 ALU(Arithmetic & Logical Unit), 곱셈과 그 결과를 누적하는 MAC(Multiplier/Accumulator), 고정소수점 정규화 기능을 제공하는 Barrel Shifter, 번지연산을 하는 DAG(Data Address Generator), 프로그램의 흐름 제어를 효율적으로 하기 위한 장치(Program Sequencer, Interrupt Controller, Status and Condition Logic)를 가지고 있다. 각 부분들을 서로 독립적으로 구성되어 있어 프로그램 작성에서 명료하게 표현할 수 있다.

3.2 고정소수점 연산

신호처리 연산에서는 실수를 사용하게 된다. 따라서 어떤 연산기를 사용하든지 실수연산을 구현해야 한다. 연산기는 크게 부동소수점 연산기와 고정소수점 연산기로 나눌 수 있는데, 이것은 수치의 정규화와 이에 따른 지수의 표현을 연산기 차원에서 지원해 주는지 여부에 달려 있다. 보다 넓은 수치 범위를 표현하기 위해서는 지수부를 둔 수치의 표현을 제공하는 부동소수점 연산

기가 바람직하다. 그러나 구조가 복잡하므로 연산기의 제작 비용과 전력 소모가 크다는 문제가 있다. 이런 이유로 고정소수점 연산기가 사용된다.

고정소수점 연산기는 실수의 표현에서 지수부를 따로 두지 않고, 일정 위치에 소수점이 있다고 가정하고 사용한다. 이 경우 발생하는 두 가지 문제점은, 수치의 표현범위의 한계와 연산 후의 소수점 위치 변동이다. 먼저, 수치표현범위의 한계에 대한 문제는 정밀도와 수치표현범위의 적절한 타협으로써 해결할 수 있다. 두 번째, 소수점 위치 변동의 문제는 수치를 1.15 형식으로 정규화 함으로써 해결할 수 있다.

연산기 내부에서 수치의 표현은 정밀도의 문제이다. 즉, 수치의 크고 작음이 문제되는 것이 아니라 얼마나 많은 가지 수의 수치를 표현할 수 있는가 하는 문제이다. 10가지의 수치를 표현할 수 있다면 1 ~ 10까지 1씩 차이가 나도록 지정할 수도 있고, 10~100까지 10씩 차이가 나도록 지정할 수도 있다. 전자에 비해 후자가 수치 범위는 10배 증가했지만 정밀도는 10배 감소했다.

곱셈이나 나눗셈 후에는 소수점의 위치가 변하게 된다. 그러나 고정소수점 연산을 위해서는 항상 소수점의 위치가 일정하게 유지되어야 올바르게 수치를 해석할 수 있기 때문에 연산 후에 항상 보정을 해 주어야 한다. 그러나 매번 자리 보정을 하는 것은 쉽지 않다. 프로그램을 작성하기도 어려울 뿐더러 연산량도 증가하게 된다. 그러나 최대 절대치를 1로 정규화 하는 1.15 형식이나 1.31형식을 사용하는 경우 소수점의 표현과 정규화가 용이하여 주로 사용된다. 이 형식은 곱셈이나 나눗셈 후 1비트의 자리 변화만이 생기기 때문에 쉽게 보정할 수 있다. ADSP-2181은 1.15형식에서 연산 결과에 대하여 자리 보정을 자동적으로 해 주는 기능을 제공한다. 1.15 고정소수점 표현 형식은 수치의 최대값을 1로 정규화 하는 것으로 표현 범위는 (-1) ~ $(1-2^{-15})$ 까지 이다. 이것은 16비트 2의 보수로 표현된 정수를 2^{15}으로 나눈 것과 같다.

1.15형식의 곱셈 결과는 1.31형식으로 확장되는데, 최종 연산 결과를 저장할 때는 16비트 1.15 형식으로 저장하게 된다. 이 경우 1.31 형식을 1.15 형식으로 변환하는 것이 필요하다. 가장 간단한 방법은 하위 16비트를 제거해 버리는 것이다. 그러나 보다 정밀도를 높이기 위해, 단순히 버리지 않고 2^{-16} 자리에서 반올림을 실시하는데, 반올림 할 때 2^{-16} 인 값은 정확히 중간 값이므로 버리는 것과 반올림하는 것 모두 가능하다. 이것에 대한 처리 방법으로 두 가지가 쓰인다. 하나는 무조건 반올림을 하는 것이고, 다른 하나는 한번은 올림을 하고 한번은 버리는 방식이다. 전자를 biased rounding이라 하고 후자를 unbiased rounding이라고 한다. 후자가 더 바람직하며 ADSP-2181은 프로세서 차원에서 이 두 기능을 모두 지원한다.

실수의 표현 방법은, 지수부와 가수부를 두는 방법과 지수부를 두지 않고 소수점을 일정 위치에 고정시키는 두 가지 방법이 있다. 부동소수점 방식에서는 연산의 결과를 항상 정규화 하여 지수부와 가수부를 갱신한다. 반면에 고정소수점 방식에서는 지수부 없이 일정 위치에 소수점을 둔다. 전자는 매 연산 결과마다 수치 보정을 해야 하는 문제가 있고, 후자는 수치의 표현 범위가 좁다는 문제가 있다. 이 두 단점을 보완하기 위해 구간(벡터) 단위로 지수를 지정하는 방법(block floating-point)이 있다. 이것은 신호처리에서의 연산이 구간(프레임) 단위로 이루어진다는 특성을 이용한 것이다. 이것은 한 구간내의 각 수치들이 제한된 수치범위 내에 존재하거나, 범위를 초과하더라도 연산 결과에 무시할만한 영향을 준다는 가정이 필요하다. 따라서 가능한 한 수치범위가 좁은 연산이나 알고리즘을 사용해야 한다.

수치표현범위와 정밀도의 문제는 연산속도와 밀접한 관계가 있다. 만약 연산속도가 충분하다면 부동소수점 방식으로 연산을 하거나 충분한 정밀도로 연산을 할 수 있기 때문이다. 그러나 가

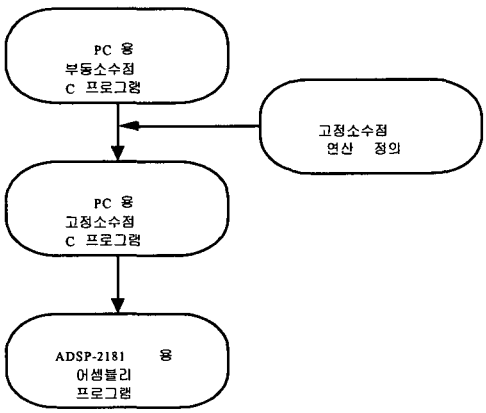
능한 한 낮은 성능을 사용하는 것이 비용 문제상 유리하므로 경우에 따른 최적의 연산을 행해야 한다. 즉, 각 연산에서 필요한 정밀도와 수치표현범위를 결정해야 하는데, 이론상 실제 구현에서 발생하는 유한 단어길이 효과에 의한 잡음을 없애기 위해서는 이 잡음이 양자화 잡음 내에 존재하도록 정밀도를 유지해야 한다. 따라서 고정소수점으로 구현하기 위해서는 먼저 알고리즘과 연산의 특성을 알아야 한다. 분석하는 파라메터의 특성과 양자화 잡음 등을 고려하여 연산 정밀도를 결정하고, 연산과정에서 수치의 범위를 파악하여 충분한 대처를 해야 한다. 특히 곱셈 결과를 누적하는 연산에서는 수치의 범위를 초과할 수 있기 때문에 연산 전에 발생할 수 있는 수치 범위만큼 소수점을 이동(나눗셈을 실시)시킨 뒤 연산을 해야 한다. 물론 연산 후에는 이때 발생한 지수의 변화를 고려해야 한다. 특히, 에너지를 비교하는 연산 등에서는 상당히 큰 수치 범위를 갖기 때문에 부득이 지수를 따로 지정해야 하는 경우도 있다.

3.3 부호화기의 구현

부호화기의 성능은 설계구조에 의해 결정되지만, 구현에서 발생하는 오차에 의해서도 큰 영향을 받는다. 따라서 추가 잡음을 없애려면 구현 오차가 양자화 잡음 내에 존재하도록 충분한 정밀도를 유지해야 한다. 그러나 이러한 문제는 연산기의 성능에 크게 좌우되므로 잡음이 일부 추가되는 것을 배제할 수 없는 경우도 있다.

효율적으로 개발을 진행하기 위해서는 개발과정을 신중하게 고려해야 한다. 본 논문에서는 그림 3과 같은 개발과정을 선택했다. DSP용 프로그램은 성능의 문제 때문에 어셈블리로 개발하기로 결정하고, 먼저 PC용 부동소수점 C 언어 프로그램으로 알고리즘의 검증과 수정 및 최적화를 실시한다. 그 다음 부동소수점 연산을 고정소수점 연산으로 대치한다. 고정소수점 연산은 사용할 연산기가 제공하는 연산을 시뮬레이션 하도록 정의한다. 연산의 대치 과정에서 유한 단어길이 효과가 발생하므로 고정소수점 연산기법을 이용한다. 알고리즘의 특성에 기반하여 정밀도와 수치범위(보호비트; guard bit)를 선택하도록 한다. 최종 목표가 DSP용 어셈블리 프로그램의 작성이기 때문에 이 과정에서 가능한 한 목표 연산기의 특성에 맞도록 구조를 설정해야 한다. 마지막으로 두 번째 단계에서 작성한 프로그램을 DSP 어셈블리로 일대일 포팅(porting)하여 완성한다. 일대일 포팅을 함으로써 어셈블리 프로그램의 검증과정에서 PC에서의 결과와 비교함으로써 정상동작 여부를 쉽게 알 수 있다. 이렇게 진행하기 위해서는 C 프로그램을 작성할 때, 목표 DSP 프로그램의 모듈 구성을 미리 고려하여 독립적인 비교가 가능하도록 작성해야 한다.

그림 3. DSP 프로그램 개발과정



IV. 구현 결과

구현 과정에서 단계적으로 검증을 거쳐 진행했기 때문에 시행착오를 줄일 수 있었다. 뿐만 아니라 DSP 고정소수점 연산을 시뮬레이션한 C 프로그램을 미리 작성하여 검증을 거친 후, 일대일 포팅 작업과 개별 모듈 단위의 비교 검증을 실시했기 때문에 연산 결과가 PC상에서 시뮬레이션한 결과와 같았다. 그러나 배정밀도 연산은 단정도 연산에 비해 3배 가량의 연산량을 필요로 하기 때문에 실시간을 위한 연산기의 성능을 훨씬 초과하는 연산량을 보였다. 주요 분석 부분의 연산량은 표 2와 같다. 이것은 배정밀도인 32비트로 모든 연산을 구현한 경우의 결과이다.

표 2. G.723.1부호화기를 32비트 연산으로 구현한 경우의 연산량

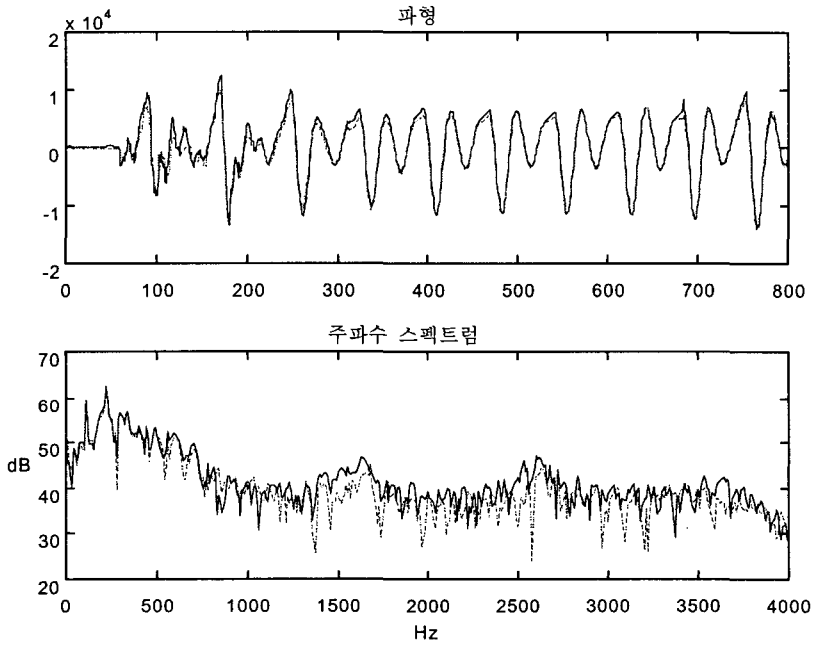
DSP 구현 / 알고리즘	최적화 전	최적화 후
LPC 분석	1.6 MIPS	1.1 MIPS
LSP 양자화	3.2 MIPS	1.9 MIPS
피치검출	3.5 MIPS	2.5 MIPS
피치예측	9.1 MIPS	5.3 MIPS
잔류신호 부호화 (MP-MLQ)	33.9 MIPS	10.6 MIPS
기타	3.7 MIPS	3.7 MIPS
전체	55.0 MIPS	25.1 MIPS

ADSP-2181은 33 MIPS의 연산성능을 가지고 있다. 따라서 실시간으로 구현하기 위해서는 약 22 MIPS 가량의 연산량을 감소시켜야 한다. 연산량을 감소시키는 가장 쉬운 방법은 정밀도를 단정도로 낮추는 것이다. 단정도로 구현하는 경우 20 MIPS 가량 소요되는 것으로 알려져 있는데, 나머지 연산능력을 정밀도를 증가시키는데 이용할 수 있다. 본 논문에서는 음질을 고려하여 32비트로 구현하고 연산량을 줄이기 위해 부분적으로 단정도로 수정하는 방법을 선택했다. 합성필터계수는 불안정한 경우 음질에 치명적이며 전체 연산량에 비해 적은 부분을 차지하므로 32비트 배정밀도를 유지하도록 하였다. 주로 연산량이 많은 잔류신호의 부호화 부분과 피치예측 부분에 대하여 연산량 감소를 실시했는데, 특히 연산량이 가장 많은 잔류신호의 부호화 부분은 1/2 ~ 2/3 가량의 연산량을 갖기 때문에 집중적인 연산량 감소를 실시했다. PC상에서 시뮬레이션을 통해 정밀도를 낮추어 연산하기 위한 연산 알고리즘을 결정하고 DSP 프로그램을 수정하였다. 그 다음으로 많은 연산을 필요로 하는 피치 예측 부분에 대하여도 마찬가지의 과정을 통해 연산량을 줄였다. 그 외의 부분은 코드 최적화만을 실시하였다. 최적화를 통해 연산량이 25 MIPS 정도로 감소하였으며 이에 의한 음질의 저하는 표 3과 같았다. 피치예측 부분은 음질에 민감하기 때문에 주의를 기울여야 한다. 잔류신호의 부호화 부분은 정밀도에 대하여 피치예측 부분에 비해 상대적으로 음질 저하가 작다. 따라서 집중적으로 최적화를 실시하여 표 2와 같이 상당히 많은 연산량을 줄일 수 있었다. [표 2]에 나타낸 연산량은 DSP 상에서 40 프레임의 연산을 수행한 결과에 대한 평균 연산량을 표기한 것이다.

표 3. G.723.1 부호화기의 구현방법에 따른 음질 비교

구 현 방 법	SNR (dB)	SegSNR (dB)
단정도 부동소수점 (32비트)	16.10	15.11
배정도 고정소수점 (32비트)	16.07	14.98
DSP 실시간 부호화기(전체 최적화)	15.72	14.66
피치예측부분만 16비트 연산으로 구현한 경우	15.96	14.86
잔류신호부분만 16비트 연산으로 구현한 경우	16.01	14.96

그림 4. 입력신호(실선)와 DSP에서 부호화를 거친 신호(점선)의 파형과 주파수 스펙트럼



실시간으로 구현된 부호화기는 표 3의 결과와 같이 상당히 우수한 음질을 유지한다. 연산량 감소를 위해 정밀도를 낮추어 연산하는 과정에서 음질이 다소 저하되었으나 원음에 근접한 성능을 보였다. 원음과 비교한 결과는 그림 4와 같다. 최적화 과정에서 연산량을 많이 차지하는 피치예측 부분과 잔류신호 부호화 부분은 연산 정밀도를 낮추어 계산하였다. 그 외의 부분은 코드 최적화만을 실시했기 때문에 음질에 영향을 미치지 않았다. 표 3은 구현 방법에 따른 음질을 비교한 것이다. 그 중 연산량을 많이 차지하는 피치예측 부분과 잔류신호 부호화 부분은 정밀도를 낮추어 연산을 실시하였다. 따라서 다소의 음질 저하가 있었다. 표 3은 DSP 실시간 부호화기를 구현하기 위해 구현 단계별 음질을 비교한 것이다. DSP 실시간 부호화기의 음질은 전반적인 최적화의 결과에 대한 음질이고, 피치예측과 잔류신호에 대한 음질은 각각 개별적으로 최적화를 실시했을 때 음질에 미치는 영향을 평가한 것이다.

DSP를 이용하여 개발할 때 사용하는 메모리 크기도 중요한 의미를 지닌다. DSP는 프로세서에 내장된 메모리를 가지고 연산을 수행해야 최대한의 성능을 발휘할 수 있기 때문이다. 본 논문

에서도 메모리에 대하여 많은 고려를 했는데, 특히 G.723.1은 벡터 양자화를 위한 테이블의 크기만으로도 9 Kword 가량이 필요하다. ADSP-2181이 사용 가능한 프로그램 메모리는 16 Kword이므로 테이블이 차지하는 메모리를 제외하면 7 Kword가 프로그램을 담을 수 있는 크기이다. 복잡한 음성 부호화기의 알고리즘을 7 Kword 내에 담아야 하므로 구현 가능성을 먼저 파악해야 한다. 현재 구현 결과 필요한 프로그램 메모리는 표 4와 같다. 표에서 기타에 해당하는 것은 지각가중필터와 전체 루틴에서 사용하는 서브루틴을 구현하는데 사용된 메모리 크기이다.

ADSP-2181에서 데이터 메모리는 16Kword가 내장되어 있으며, 대부분의 음성부호화기에서 데이터 메모리는 프로그램 메모리에 비해 적은 크기를 필요로 하므로 충분하다. 따라서 데이터 메모리에 대한 고려는 특별히 하지 않았다. 사용된 데이터 메모리는 5071 word이다.

표 4. G.723.1 부호화기의 메모리 사용현황(프로그램 메모리)

알 고 리 즘	사용 메모리(word)
메인 모듈	524
LPC 분석	365
LSP 양자화	1050
피치검출	217
피치예측	435
잔류신호 부호화 (MP-MLQ)	771
테이블	8858
기타	760
전체	12,980

V. 결 론

음성부호화기를 고정소수점 DSP를 이용하여 개발하는 것은 상당히 복잡한 작업이다. 특히 고정소수점 연산을 효율적으로 수행하지 않으면 치명적인 음질 저하가 생긴다. 또한 제한된 성능의 연산기를 가지고 실시간 연산을 하기 위해서는 적당한 성능저하를 감수해야 하는 경우도 발생한다.

본 논문에서는 G.723.1음성부호화기를 ADSP-2181을 이용하여 실시간으로 동작하도록 제작하면서 고정소수점 DSP를 이용하여 구현하는 방법을 제시하였다. 특히 고정소수점 연산방법과 문제점 및 고정소수점 연산기용 프로그램을 개발하는 과정을 제시했다. 뿐만 아니라 G.723.1 부호화기를 32비트 배정도로 구현하여 연산량을 분석하였고, 실시간으로 동작시키기 위해 연산량을 감소시키는 방법과 연산량 감소 후의 음질의 차이를 비교하였다. 최종적으로 실시간으로 동작하는 부호화기의 음질을 평가하고 원음과 비교하였다. G.723.1 부호화기는 음질 향상을 위해 다양한 기법이 도입되어 타 부호화기의 제작에 참고가 될 것이다. 본 논문에서 제작한 부호화기는 실시간으로 동작하며, 음질이 우수하여 실제 응용에 다양하게 사용될 것으로 전망된다.

참 고 문 헌

[1] 손종서 · 김시현 · 강지양 · 성원영. 1996. "FS-CELP 음성부호화기의 고정소수점 성능 분석 및 구현." *한국통신학회논문지*, 21(2), 365-374.

[2] Juin-Hwey Chen & Allen Gersho. 1995. "Adaptive Postfiltering for Quality Enhancement ofCoded Speech." IEEE Trans. *On Speech and Audio Processing,* 3(1), 59-71.

[3] Peter Kabal & Ravi & Prakash & Ramachandran. 1986. "The Computation of Line Spectral Frequencies using Chebyshev Polynomials." IEEE Trans. *On Acoustics, Speech and Signal Processing,* ASSP-34(6).

[4] S. Cucchi & M. Fratti & M. Ronchi. 1996. "On Approving Performance of Analysis by Synthesis Speech Coders." IEEE Trans. *On Speech and Audio Processing,* 4(3).

[5] Ravi P. Ramachandran & Peter Kabal. 1989. "Pitch Prediction Filters in Speech Coding." IEEE Trans. *On Acoustics, Speech and Signal Processing,* 37(4).

[6] W.B. Kleijn. 1994. "On the Periodicity of Speech Coded with Linear-Prediction Based Analysis by Synthesis Coders." IEEE Trans. *On Speech and Audio Processing,* 2(4), 539-542.

[7] 장동일. 1993. *스펙트럼 정보를 이용한 CELP 음성부호화기의 계산량 감소에 관한 연구.* 서울대학교 석사논문.

[8] ALALOG DEVICES. 1992. "Digital Signal Processing Applications Using the ADSP-2100 Family." 1, 13-50, *ANALOG DEVICES.*

[9] A. M. Kondoz. 1994. "Digital Speech Coding for Low Bit Rate Communication Systems." 160-174, *WILEY.*

[10] W. B. Kleijn & K. K. Paliwal. 1995. "Speech Coding and Synthesis." 51-77, *ELSEVIER.*

접수일자 : '98. 2. 28.
게재결정 : '98. 4. 2.

▲ 박정재
강원도 춘천시 효자2동
강원대학교 공과대학 전자공학과(우 : 200-701)
e-mail: jjpark@dsplab.kangwon.ac.kr

▲ 정익주
강원도 춘천시 효자2동
강원대학교 공과대학 전자공학과(우 : 200-701)
Tel: (0361) 250-6322 (O) Fax: (0361) 56-6327
e-mail: ijchung@cc.kangwon.ac.kr