논문97-2-1-02

실시간 MPEG-1 오디오 인코더의 설계 및 구현

전 기 용*, 이 동 호**, 조 성 호***

A Design and Implementation of the Real-Time MPEG-1 Audio Encoder

Ki Young Jeon*, Dong Ho Lee**, and Sung Ho Cho***

요 약

본 논문에서는 하나의 TMS320C31 Digital Signal Processor (DSP)를 사용하여 실시간으로 동작하는 Motion Picture Experts Group-1 (MPEG-1) 오디오 인코더 시스템을 구현하였다. 우선 MPEG-1 Audio Layer-2 및 심리음향모델-1 관련 기본 알고리듬을 C-언어로 구현하여 기본 동작을 확인하였다. 그리고 전체 실행 시간을 줄이기 위하여, 이를 다시 Texas Instruments (TI) 어셈블리어로 작성하였다. 마지막으로, MPEG-1 오디오 인코더 시스템을 위한 실제 DSP 하드웨어 회로 보드를 설계, 제작하였다. Analog-to-Digital Converter (ADC) 제어, 입출력 제어, 그리고 DSP 보드에서 PC로의 비트열 전송과 같은 주변 모듈들은 Very High Speed Hardware Description Language (VHDL)을 사용하여 Field Programmable Gate Array (FPGA)로 구현하였다. 제작된 시스템은 48 kHz로 샘플링 되는 스테레오 오디오 신호를 실시간으로 처리하여 192 kbps 비트율로 부호화된 비트열을 출력시킨다. 다양한 형태의 스테레오 오디오 신호를 통해, 제작된 오디오 인코더 시스템의 실시간 동작과 양질의 오디오 신호가 복원됨을 확인하였다.

Abstract

In this paper, a real-time operating Motion Picture Experts Group-1 (MPEG-1) audio encoder system is implemented using a TMS320C31 Digital Signal Processor (DSP) chip. The basic operation of the MPEG-1 audio encoder algorithm based on audio layer-2 and psychoacoustic model-1 is first verified by C-language. It is then realized using the Texas Instruments (TI) assembly in order to reduce the overall execution time. Finally, the actual DSP circuit board for the encoder system is designed and implemented. In the system, the side-modules such as the analog-to-digital converter (ADC) control, the input/output (I/O) control, the bit-stream transmission from the DSP board to the PC and so on, are utilized with a field programmable gate array (FPGA) using very high speed hardware description language (VHDL) codes. The complete encoder system is able to process the stereo audio signal in real-time at the sampling frequency 48 kHz, and produces the encoded bit-stream with the bit-rate 192 kbps. The real-time operation capability of the encoder system and the good quality of the decoded sound are also confirmed using various types of actual stereo audio signals.

* LG 정보통신 전송연구단 Transmission Division, LG Information & Communications, Ltd.
** 한양대학교 제어계측공학과 Dept. of Control and Instrumentation Engineering, Hanyang University
*** 한양대학교 전자공학과 Dept. of Electronic Engineering, Hanyang University
※ 본 논문은 한양대학교 공학기술연구소와 전자재료 및 부품연구 센터의 부분적인 연구비 지원에 의한 결과임.

Ⅰ. 서 론

디지털 오디오는 80년대에 들어서면서 CD나 DAT와 같은 대용량 저장 매체의 개발과 함께 다양한 오디오 기기의 표준으로 점차 자리를 잡아가고 있다. 그러나 디지털 오디오 데이터는 많은 정보량을 가지므로 공중파 방송에서와 같이 제한된 대역폭을 갖는 매체에서 사용하기 위해서는 그 데이터 정보의 압축이 필수적이다. 이에 80년대 후반부터 미국의 AT&T, Dolby Lab, 유럽의 IRT, Philips와 일본의 Sony등의 여러 연구소에서 고음질 오디오 압축 기술을 개발한 바 있는데, 이와 같은 기술은 공통적으로 기존의 데이터 압축 기법에 사람의 청각 특성을 고려하여 결합한 형태를 갖는다 [1][2].

실제적인 부호화 기술의 대표적인 것으로, 동영상과 그에 부가되는 오디오 압축 방식에 대해 International Standard Organization (ISO)에서 규정한 MPEG 규격이 있다. MPEG-1은 약 1.5 Mbps에서 동영상과 오디오를 압축할 수 있는 부호화 방식으로서 Masking-Pattern Adapted Universal Subband Integrated Coding and Multiplexing (MUSICAM) 방식을 사용하였다. 이것은 고품질 디지털 방송을 위해 6 Mbps이상의 전송율을 갖는 다채널 구조의 MPEG-2로 확장되었다. 현재 MPEG-1 및 MPEG-2는 고음질 오디오와 고화질 비디오 압축 방식의 국제 규격으로 채택되었으며 [3][4], 이와 같은 규격에 맞추어 오디오 및 비디오 신호를 실시간 압축하는 하드웨어 구현을 위한 움직임이 매우 활발히 진행되고 있다.

본 연구에서 구현한 MPEG-1 오디오는 인간의 청각 특성을 이용하여 주관적인 음질의 손상 없이 높은 압축율을 가지면서, 동시에 CD 수준의 음질을 얻을 수 있도록 제작되었다. 특히, 한 개의 범용 DSP Chip을 사용하여 [6] 실시간MPEG-1 오디오 부호화 시스템을 구현하였다. 구현된 부호화기는 Layer-2에서 심리음향모델-1을 사용하였다 [3]-[5]. 제작된 DSP 보드에 오디오 신호를 입력시키면 DSP 프로그램은 이 신호를 실시간으로 처리하여 압축된 MPEG-1 오디오 비트열을 출력시킨다. 출력된 비트열을 PC로 입력받아 상용화된 프로그램을 이용하여 복호 해 보았으며, 부호화기가 실시간으로 잘 동작하고 양질의 오디오 신호가 복원됨을 확인하였다.

본 논문의 구성은 다음과 같다. 2장에서는 MPEG-1 오디오의 알고리듬을 소개하고 이의 실시간 구현 방법에 대해 논하였다. MPEG-1 오디오 규격에 제시된 주요 기본 알고리듬 가운데 서브밴드 분석과 심리음향모델, 그리고 비트할당 알고리듬에 대해 간단히 설명하였다. 실시간 DSP 어셈블리어 프로그램 과정에서의 중요한 사항과 알고리듬 최적화 방안은 3장에서 논의되었다. 4장에서는 개발된 DSP 하드웨어 특성과 성능 평가에 대하여 설명하였다. 그리고 마지막으로 5장에서 결론을 맺었다.

Ⅱ. MPEG-1 오디오 알고리듬의 실시간 구현

이제 MPEG-1 오디오 부호화기의 전체 구성과 각 알고리듬의 기능에 대해 간단히 소개하고, 이의 실시간 구현 방법에 대해 살펴본다.

1. MEPG-1 오디오 부호화기의 전체 구성

MPEG-1 오디오 부호화기의 알고리듬은 (그림 1)에 도시한 바와 같이, 크게 네 개의 블럭으로 이루어진다 [3]-[5]. 첫 번째 블럭은 입력되는 16비트 오디오 데이터를 주파수 영역에서 등 간격으로 나뉘어진 32개 서브밴드 상의 데이터로 변환시키는 서브밴드 분석 블럭이고, 두 번째 블럭은 입력되는 데이터로부터 부호화 해야 할 각 서브밴드의 음압레벨을 결정하는 심리음향모델 블럭이다. 세 번째 블럭은 각 서브밴드의 데이터를 부호화하는데 필요한 비트 수를 결정하고, 결정된 비트 수에 따라 데이터를 양자화시키는 비트 할당 및 양자화 블럭이며, 마지막으로 네 번째 블럭은 적절한 헤더와 부호화한 데이터에 관련된 정보를 양자화된 데이터에 덧붙여 비트열을 완성시키는 비트열 형성 블럭이다.

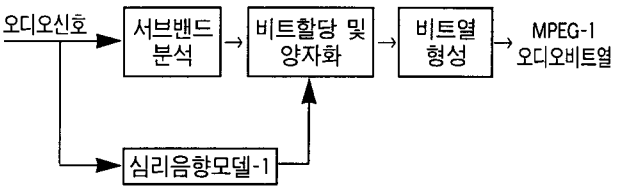

그림 1. MPEG-1 오디오 부호화기 블럭도
Fig. 1. A block diagram of the MPEG-1 audio encoder

MPEG-1 오디오 부호화기를 실시간으로 구현하기 위한 최소한의 성능을 생각하여보면 다음과 같다. 48kHz 샘플링 주파수의 오디오를 MPEG-1 부호화 알고리듬으로 실시간 처리하기 위하여는, 샘플링 되어 DSP 보드로 입력되는 오디오 데이터의 샘플 수가 2304개 되기 전에, 앞서 입력된 2304개의 데이터가 모두 부호화되어 비트열로 출력되어야 한다. 이때 2304개의 데이터는 1152개의 데이터가 동시에 스테레오로 들어오는 경우이므로, 실제 2304개의 데이터가 입력되는 시간 간격은 1152개의 데이터가 입력되는 시간 간격과 동일하며, 따라서 1152/48000=24 msec로 주어진다. 그러므로, 실시간 처리를 위해서는, 24 msec 이내에 앞서 입력된 2304개의 스테레오 데이터들이 모두 부호화되어 비트열로 출력되어야 한다.

사용된 TMS320C31 DSP chip은 50 MHz clock으로 동작되는데 실제 DSP chip은 이 클럭을 2분주하여 사용한다. 따라서 DSP chip의 동작 사이클은 $1/(25\times10^6)$sec가 된다. 이때 Assembly 프로그램은 1개의 명령어를 수행하는데 2

분주 된1개의 클럭이 필요하므로, 24 msec 동안 DSP 명령어는 $0.024\times25\times10^6=600{,}000$번 수행된다. 즉, 600,000개의 명령어 이내에 오디오 부호화 알고리듬이 수행되도록 알고리듬과 DSP 어셈블리어 프로그램을 최적화 해 주어야 실시간으로 동작이 가능하다.

2. 서브밴드 분석

MPEG-1 오디오 부호화 알고리듬 가운데 첫번째 과정인 서브밴드 분석 블럭에서는 Nyquist 주파수 f_N의 주파수 범위를 같은 간격의32개 서브밴드로 나누고, 각 서브밴드의 데이터를 독립적으로 부호화한다. 이에 따라, 입력되는 데이터는 주파수 영역에서 32개 서브밴드 내의 데이터로 변환된다. 먼저, 16비트 데이터는 한번에 32개씩 서브밴드 분석 블럭으로 입력된다. 데이터들이 서브밴드 분석 알고리듬을 통과되면 32개 서브밴드에 데이터가 각각 1개씩 결정된다. 만약 MPEG-1 오디오 Layer-2에서 한 채널만을 고려할 때, 한 패킷 당 부호화 해야 할 데이터 샘플 수가 1152개이고, 1152개의 서브밴드 분석된 데이터를 얻기 위해서는 서브밴드 분석 블럭을 36회 수행하여야 한다.

본 연구에서는 스테레오 채널을 처리하므로 필요한 서브밴드 분석된 데이터의 수는 2304개이다. 따라서, 72회 서브밴드 분석 블럭이 수행되어야 한 패킷의 부호화 대상 데이터를 얻을 수 있게 된다. 이와 같은 계산량의 부담으로, 서브밴드 분석 블럭이 보통 실시간 처리의 가장 난해한 블럭이다. 실제로 이 블럭을 처리하기 위하여는 서브밴드 분석 블럭의 동작을 최적화하여야 하며, 동시에 어셈블리어가 제공하는 다양한 실시간 처리 능력을 활용하여야 한다.

서브밴드 분석 블럭의 최적화에서 계산상의 가장 큰 부담은, 식(1)에 주어진 코사인 함수 M_{ik}를 사용하여 각 서브밴드의 데이터 한 개를 얻기 위해 64회의 곱셈과 63회의 덧셈을 수행하는데 있다. 이와 같은 실수 연산을 각각 32회 수행하여야 전체 서브밴드의 데이터를 얻을 수 있으며, 한 패킷의 데이터를 얻기 위하여는 이와 같은 계산을 다시 72회 계속하여야 한다.

$$M_{ik}=\cos\left[\frac{(2i+1)\cdot(k-16)}{64}\cdot\right]\quad 0\le i\le 31, 0\le k\le 63 \qquad (1)$$

이 같은 연산을 표현 그대로 수행하면 계산상에 큰 부담이 되며 실시간 처리가 매우 어렵게 된다. DSP 연산에 가운데 가장 큰 부담은, 한번 계산마다 곱셈을 수행한 후 그 값을 저장하고, 다음 값들에 대하여도 동일하게 처리한 후, 저장된 값들을 다시 불러내어 덧셈을 수행하여야 하는 과정이다.

설명을 위해 다음 예를 생각해 보자. 계산으로 구해진 임의의 중간 값에 (이를 x라 하자.) cosine 함수 값을 (이를 α라 하자.) 곱하여 저장한 후, 또 다른 중간 값을 (이를 y라 하자.) 구하여 α나 $-\alpha$를 곱한 다음, 앞에서 저장한 값을 다시 불러내어 더한다고 하자. 이와 같은 경우, 다음과 같은 방법을 사용하면 계산량을 상당히 줄일 수 있다. 즉, 실제 cosine 함수 값 M_{ik}는 크기가 같은 값들이 부호가 같거나 반대의 부호를 갖는 반복되는 규칙성을 가지고 있다. 따라서 이와 같은 규칙성을 이용하여 실제로 같은 크기의 값들이 중복되지 않도록 반드시 필요한 값들 만을 미리 구하여 SRAM에 저장하여 둔다. 그리고 최종적인 알고리듬은 동일하지만 중간 계산상의 알고리듬을 조금 변경하여 임의의 중간 값 x와 y를 연속하여 구한 후, 두 값을 더하거나 (x와 y에 동일한 α를 곱한 후 더한 경우) 뺀 다음 (x와 y에 반대 부호의 α를 곱한 후 더한 경우) cosine 함수 값 α를 한번만 곱해준다. 이와 같은 알고리듬의 변경은 곱셈의 횟수와 중간 계산 값들을 SRAM에 일단 저장한 후 매번 다시 불러내어야 하는 계산상의 부담을 크게 줄여준다.

계산상의 부담을 줄이기 위한 또 하나의 방법은 서브밴드 분석 알고리듬에서 가장 처음에 나타나는 블럭에 적용된다. 즉, 입력되는 데이터를 최신의 것들로만 선택하여 512개의 배열을 유지하는 부분이다. 이 부분에 대하여 대개의 경우에는 회전 큐를 사용하는데, 이 회전 큐를 사용할 때에는 큐의 가장 마지막 단을 항상 확인해야 하며, 이로 인하여 어셈블리어 상에 계속해서 비교문이 들어가게 된다. 이것보다는 1630개의 확장된 배열을 잡아서 처리하는 것이 계산상의 부담을 많이 덜어준다. 1630개의 배열 가운데 가장 마지막 480개에는 최초의 프로그램 실행 시 모두 0.0을 저장해두고, 입력되는 32개씩의 데이터들을 그 상위 어드레스 장소에 역순으로 (즉, 낮은 어드레스 방향으로) 차례로 계속해서 저장한다. 그 후, 이 배열과 규격에서 이미 제시된 window함수가 1:1로 곱해지게 되는데, 이 과정에서 비교문의 사용을 피할 수 있게 된다.

한편, MPYF||LDF 및 MPYF||ADDF 등과 같은 병렬처리 명령어를 적재적소에 사용한다면 더욱 빠른 계산이 가능하게 된다.

3. 심리음향모델-1 (Psychoacoustic Model-1)

심리음향모델-1에서는 각 서브밴드 내의 최대 음압레벨(Maximum Sound Pressure Level)과 최소 마스킹 역치(Minimum Masking Threshold)를 결정하여, 해당 서브밴드의 데이터를 부호화해 주기 위한 음압레벨을 구한다. 여기에서 최대 음압레벨이란 각 서브밴드의 데이터를 주파수 영역에서 보았을 때 최대의 값을 가지는 음압을 말하며, 최소 마스킹 역치란 마스커 (Masker)에 의해 임의의 주파수 위치 주위에서 마스킹되는 최소의 음압레벨을 말한다. 부호화해 주기 위한 음압레벨은 각 서브밴드 마다 결정된 최대 음압레벨에서 최소 마스킹 역치를 감하여 결정되는데, 이 값이 해당 서브밴드의 소위 Signal-to-Mask Ratio (SMR) 이다.

MPEG-1 오디오 부호화 알고리듬의 Layer-1과 Layer-2

에서는 심리음향모델-1을 사용하고, Layer-3에서는 심리음향모델-2를 사용한다. 심리음향모델의 입력은 서브밴드 분석으로 입력된 16 비트 스테레오 데이터들과 동일하며, 출력은 각 서브밴드 당 SMR이다.

심리음향모델-2의 구현상에 가장 큰 부담은 다음의 두 가지로 요약할 수 있다. 첫 번째 부담은 $\log_{10}X$및 10^X등과 같은 대수함수의 어셈블리어 구현이다. 대수함수를 구현함에 있어서 계산상의 정밀도와 계산하여야 할 명령어의 수 사이에는 상호 이율배반적인 관계가 형성된다. 즉, 정밀도를 높이면 계산량이 늘어나고, 반대로 계산량을 줄이면 정밀도가 떨어지게 된다. 이번에 구현된 오디오 인코더에서는 대수함수의 어셈블리어 프로그램 시, X값을 입력하고 $\log_{10}X$ 및 10^X를 구하는데, 허용오차의 한계를 고려하여 각각 약 40개의 클럭이 소요되도록 일단 정하였다.

두 번째 부담은 각 주파수 위치에서 마스킹 역치를 모두 구한 후, 각 주파수 위치를 다시 서브밴드 영역으로 매핑하여 각 서브밴드상의 최소 마스킹 역치를 구하는 과정이다. 이 경우에는 각 주파수 위치에서 마스킹 역치를 구함에 있어, 해당 주파수 범위 내의 마스커들에 의한 개별 마스킹 역치를 모두 구한 후, 이 값들로부터 해당 위치의 마스킹 역치를 구해야 한다. 실제로 우리가 필요한 값들은 각 서브밴드 상의 최소 마스킹 역치인데, 이와 같이 계산할 경우 최후까지 각 서브밴드에서 최소 마스킹 역치로 결정되지 않을 값들마저도 모두 구해야만 하므로 계산상의 부담이 매우 크다.

본 논문에서 사용된 이들에 대한 대안은 각각 다음과 같다. 먼저, 첫 번째로 언급된 $\log_{10}X$ 및 10^X 등과 같은 대수연산의 효율화를 위한 대안을 살펴보자. 이를 위해, 심리음향모델 상에서 dB로 표시된 두 음압의 값들을 각각 X와 Y로 표시하고, 이 두 값의 평균을 Z로 표시하자. 이 경우 Z는 다음과 같은 식으로 나타난다.

$$Z = 10\log_{10}\left(10^{\frac{X}{10}} + 10^{\frac{Y}{10}}\right) \tag{2}$$

이 식에서 보통은 $\log_{10}X$와 10^X라는 두 가지 대수함수를 어셈블리어로 구현하여야 한다. 그러나 예를 들어 $\log_{10}X$를 어셈블리어 프로그램으로 구현하여 이를 마크로로 정의한 후 이를 한번 호출하게 되면, 계산상의 유효 숫자에 범위를 10^{-7}으로 했을 경우에 약 40회 이상의 클럭을 필요로 한다. 그리고 10^X를 어셈블리어로 구현할 경우에도 비슷한 클럭 횟수를 필요로 한다. 따라서 식(2)를 계산하는 데만도 얼마만큼 계산상의 부담이 드는지를 짐작할 수 있다.

이에 대한 대안으로 dB영역이 아닌 power영역에서 계산을 수행한다. 식(2)에서 X, Y, Z의 power값들을 각각 x, y, z라 할 때 이들 사이에는 다음과 같은 관계가 만족된다.

$$X = 10\log_{10}X \tag{3}$$

$$Y = 10\log_{10}Y \tag{4}$$

$$Z = 10\log_{10}Z \tag{5}$$

이 식으로부터 dB영역의 Z값을 power영역으로 옮기면 다음과 같은 계산식이 성립함을 알 수 있다.

$$Z = X + Y \tag{6}$$

따라서 식(2)가 단순이 두 power 값들의 합으로 결정될 수 있음을 알 수 있다. 이와 같은 방법으로 처리하여 각 주파수 상의 power 값들로 부터 마스커들과 마스커 음압 값들을 결정하면 매우 많은 계산량을 줄일 수 있다. 그리고 최종적으로 각 주파수 위치에서의 개별 마스킹 역치를 구하는 경우에 결정된 마스커들의 음압 값들 만을 dB 값으로 변환시켜 주면 된다.

이제, 두 번째로 언급된 각 주파수 위치상의 마스크 역치를 구하는 문제에 대한 대안을 생각해 보겠다. 앞서 언급하였듯이 심리음향모델을 그대로 사용하여 각 서브밴드상의 최소 마스킹 역치를 구하는 경우에는, 마지막까지 각 서브밴드상의 최소 마스킹 역치로 선택되지 않을 주파수 위치상의 마스킹 역치도 모두 구해야 한다. 이는 매우 큰 계산상의 부담이 되므로 다음과 같은 방법을 사용하여 계산상의 부담을 줄일 수 있었다. 즉, 미리 각 주파수 위치를 서브밴드 영역으로 매핑시킨 후, 임의의 서브밴드 영역 내의 첫번째 주파수 위치에서 마스킹 역치를 구한 다음, 그 값을 해당 서브밴드의 최소 마스킹 역치로 일단 간주한다. 그 다음 주파수 위치에서는 계산 도중, 앞서의 최소 마스킹 역치 값보다 큰 값이 나오면 그 주파수 위치에서 더 이상의 계산을 수행하지 않는다. 이와 같은 방법으로 처리하면 해당 서브밴드 내의 주파수 위치에서 모든 마스킹 역치 값을 계산하지 않고 좀 더 빠르게 최소 마스킹 역치 값을 구할 수 있다.

4. 비트 할당 알고리듬

서브밴드 분석을 수행하면 각 서브밴드 마다 부호화 해야 할 데이터를 36개씩 얻는다. 그리고 심리음향모델에 의하여 각 서브밴드 마다 부호화 해야 할 음압레벨을 결정한다. 하지만 전체 2304개의 데이터를 부호화 시킬 수 있는 비트 수는 고정되어 있다. 전체 비트의 수가 고정되어 있는 반면, 각 서브밴드 마다 부호화 해야 할 음압레벨이 서로 다르기 때문에 각 서브밴드 마다 할당되는 비트 수 역시 다를 수 밖에 없다.

서브밴드 마다 데이터를 부호화하는데 사용되는 비트 수를 다르게 결정하는 기준은 심리음향모델에서 구한 SMR이다. 어느 서브밴드의 데이터가 다른 서브밴드의 데이터에 비해 큰 SMR 값을 갖는다면, SMR이 작은 서브밴드의 데이터보다 좀더 잘 부호화해 주어야 하므로 비트를 좀 더 많이 할당한다.

일반적으로, 서브밴드의 데이터 부호화에 사용되는 비

트의 수를 1 비트 증가시키면 약 6dB 정도의 양자화 에러를 줄일 수 있다. 즉 Signal-to-Noise Ratio (SNR)가 약 6dB 상승하게 된다. 이 경우 Mask-to-Noise Ratio (MNR)를 식(7)과 같이 정의한다. SNR은 비트의 수에 따라서 결정되는 값으로, [3]과 [4]에 잘 나타나 있다.

$$MNR = SNR - SMR \tag{7}$$

식(7)에 의하여 각 서브밴드의 MNR을 결정하고, 가장 작은 MNR을 갖는 서브밴드에만 한 비트를 추가 할당하여 그 서브밴드의 MNR을 상승시킨다. 그리고 그 서브밴드에 한 비트를 추가 할당 함으로서 전체 서브밴드의 데이터를 부호화하는데 필요한 비트 수를 계산한다. 필요한 비트 수가 한 패킷의 데이터를 부호화하도록 결정되어 있는 전체 비트 수 보다 작으면 다시 각 서브밴드의 MNR을 계산하여 한 비트씩 추가하는 비트 할당 알고리듬을 계속한다. 만약 부호화에 필요한 비트 수가 이미 결정되어 있는 전체 비트 수 보다 커지게 되면, 알고리듬을 종결하고 그 이전에 각 서브밴드 마다 할당되어 있는 비트 수를 이용하여 각 서브밴드의 데이터를 부호화한다.

Ⅲ. 고속 연산을 위한 DSP 처리 전략

본 논문에서 사용한 TMS320C31 DSP Chip은 고속 연산을 위한 보편적인 종래의 TMS320시리즈의 DSP와는 달리, 개량형 Harvard Architecture를 채택하고 여러 종류의 전용 버스를 가지고 있다. 특히 Direct Memory Access (DMA), serial port, timer 0, timer 1 등을 위한 parallel bus를 각각 따로 두어 복수의 버스에 의한 고속의 병렬처리가 가능하다 [6].

구현한 DSP 보드에서 프로그램의 동작은 모두 한 명령어가 2분주 된 한 개의 클럭에 동작이 되도록 하였다. 이를 DSP가 0 wait로 동작하게 하였다고 말한다. 예외로는 DSP 보드가 power-up시에 EPROM으로 부터 프로그램의 up-loading 순간인데, 이때는 7 wait으로 (즉, 2분주 된 8개의 클럭으로) 동작되며, 이는 저속의 EPROM으로 부터 프로그램의 up-loading을 확실하게 보장한다. 프로그램 up-loading 이후의 모든 동작에서 DSP는 0 wait으로 동작한다. 따라서 주변 소자들은 DSP CPU의 고속 동작에 어떤 부하(load)도 주지 않게 된다.

또한 어셈블리어 프로그램 상에서 pipeline이 훼손되지 않도록 branch문의 사용을 최대한 억제하였다. 그 대신, TMS320C31 DSP는 delayed branch라는 명령어를 가지고 있다. Delayed branch는 이 명령어 이후로 나오는 3개의 명령어를 수행하고 한 클럭 만에 원하는 번지로 프로그램 수행 포인터를 옮겨 준다. 따라서 pipeline이 훼손되지않고 효율적인 고속의 DSP 동작을 보장한다.

이와 비슷한 특성으로서 pipeline conflict를 들 수 있다. 예를 들어, 보조 레지스터AR0 ~ AR7 중에서 한 개의 레지스터에 값을 적기 위해 assess할 경우, 이 다음 명령어에서 다른 보조 레지스터의 값을 변경시키려면 pipeline이 훼손된다. 즉, 연속하여 두개의 보조 레지스터를 access할 수는 없다. 이를 pipeline conflict라고 한다. 이 이외에도 DSP의 고속 동작을 저해하는 pipeline conflict에는 여러 가지 종류가 존재한다 [6]. 어셈블리어 프로그램 작성을 하면서 고속 동작을 방해하는 pipeline conflict를 없애기 위해 부분적으로 알고리듬을 수정하였다.

그리고 한 클럭에 두개의 명령어를 수행하는 병렬처리 명령을 최대한 활용하였다. 50 MHz의 외부 클럭을 입력받아 2분주한 25 MHz의 클럭으로 동작하는 TMS320C31-50 DSP가 최고 성능으로 50 MFlops를 낼 수 있는 이유가 바로 병렬처리 명령 때문이다. 즉, 25 MHz 클럭을 사용하여 한 클럭에 두개의 부동 소수점 연산을 할 수 있기 때문에 최고 성능은 50 MFlops가 되는 것이다. 이와 같이 고속 동작에 필수적인 병렬처리 명령어의 단점은 그러나, 적용할 수 있는 레지스터들의 형식에 한계가 있다는 것이다. 따라서 병렬처리 명령을 최대한 적절하게 사용할 수 있도록 어셈블리어 프로그램 작성 도중 레지스터의 할당에 많은 노력을 기울였다.

마지막으로, DSP Chip 내부에 있는 2 kword의 내부 RAM을 적극 활용하였다. 실제로 DSP Chip은 외부 SRAM에 대하여 한 클럭에 한번 access가 가능하지만, DSP Chip 내부에 있는 레지스터와 내부 RAM은 2회 이상 access가 가능하다. 따라서 예를 들어 FFT 수행의 경우, 필요한 512개의 cosine 데이터들은 내부 RAM Block 1에 저장하였고, 입력 데이터와 출력 데이터 각각 1024개는 내부 RAM Block 0에 저장되도록 하였다. 따라서 고속의 프로그램 수행이 가능하다. 주의할 점은, 하드웨어 구현 시, TMS320C31 DSP Chip의 인터럽트 벡터가 내부 RAM 1에 존재하므로 이와 cosine 데이터가 겹치지 않도록 해야 한다.

이와 같은 TMS320C31 DSP의 고속 동작을 위한 여러 가지 특성을 살리는 데에는 무엇보다 DSP에 대한 확실한 이해가 필수적이다. 심지어 이를 위하여 프로그램 흐름도의 적절한 변화가 요구되기도 하였다.

Ⅳ. MPEG-1 오디오 부호화기 하드웨어 구현

MPEG-1 오디오 부호화기가 실시간으로 오디오를 처리한다는 의미는 입력되는 ADC된 데이터의 갯수가 2304개가 되기 이전에, 앞서 오디오 부호화 알고리듬으로 처리가 시작된 데이터들이 모두 부호화된 비트열로 출력되어야 함을 말한다. MPEG-1 오디오 부호화기를 실시간으로 동작시키기 위해 구현한 DSP 보드의 기능별 블럭도를 (그림 2)에 나타내었다. 하드웨어 구성도에서 알 수 있듯이, 전체적인 하드웨어 구성은 입력부, DSP 처리부, 그리고 출력부 등 세 부분으로 나누어진다.

입력부는 입력되는 오디오 애널로그 신호를 16비트 스테레오 채널로 A/D하여 FIFO로 전달한다. 입력부에서 사

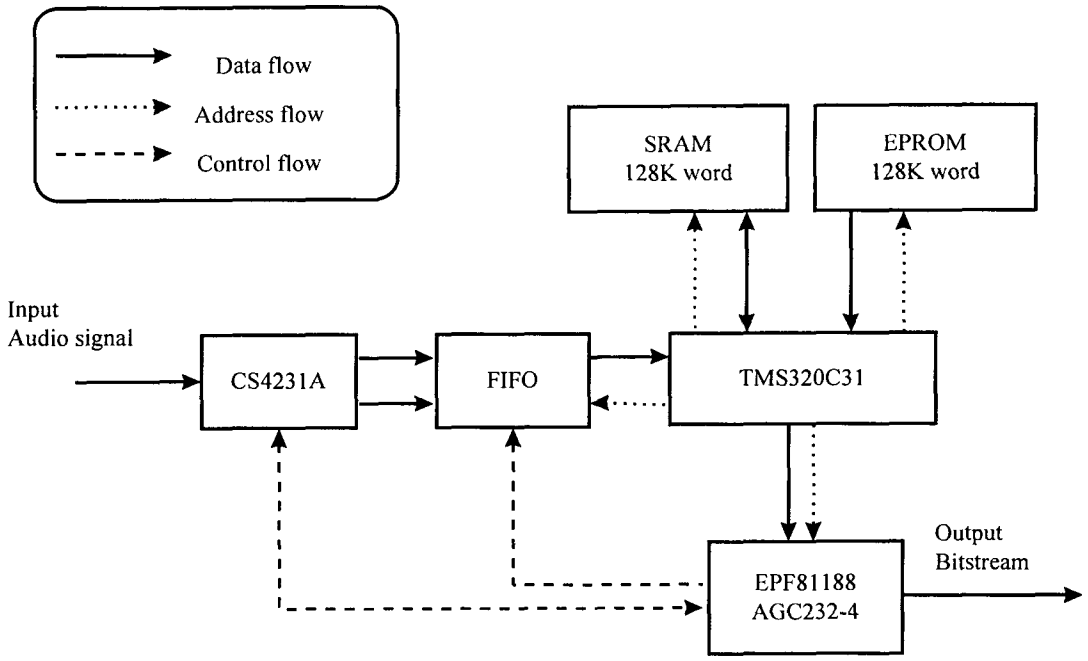

그림 2. MPEG-1 오디오 부호화기 하드웨어 구성도
Fig. 2. A hardware scheme for the MPEG-1 audio encoder

용한 ADC는 오디오 전용 코덱인 Crystal 사의 CS4231A로, 칩 내부에 CPU가 내장되어 있다. FIFO에 한 패킷의 데이터가 저장되어 있다는 사실을 DSP로 알려주는 역할은 FPGA에서 수행한다.

일단 Power-up이 되면 FPGA는 CS4231A 내의 CPU를 초기화하여 CS4231A가 48 kHz로 샘플링 된 16 비트 스테레오 채널의 데이터를 출력시키도록 한다. 그리고 CS4231A에서 출력되는 데이터들의 수를 계산하여, FIFO에 저장된 데이터의 수가 2304개가 되는 순간 FPGA는 DSP에 인터럽트를 걸어준다. DSP 처리부는 FPGA에 의해 인터럽트가 들어오면, 인터럽트 처리 루틴에서 FIFO내의 데이터를 SRAM으로 옮기고 오디오 부호화 알고리듬을 수행하기 시작한다. 그리고 최종적으로 데이터를 부호화된 비트열로 만든다.

출력부는 부호화된 비트열을 호스트 PC로 전달하는 기능을 한다. 비트열이 완성되면서 DSP CPU는 완성된 비트열을 FPGA로 전송한다. FPGA는 전송받은 비트열을 8 비트씩 나누어 FIFO로 저장시킨다. 호스트 PC는 FIFO에 접근하여 저장된 데이터를 받아간다.

부호화 알고리듬은 어셈블리어로 프로그램 되어 EPROM에 저장되어 있으며, power-up시 EPROM에서 SRAM으로 옮겨져 수행된다. 앞서 언급한ADC의 제어, ADC로부터 DSP CPU로의 인터럽트 제어, 그리고 어드레스 디코딩, 출력 데이터 비트의 정렬 모듈들은 모두 VHDL [7]-[11] 프로그램 되어 FPGA [12]로 구현하였는데, 사용한 FPGA는 Altera사의 EPF81188AGC232-4이다.

한편, 구현한 부호화기의 성능 평가는 다음과 같이 수행하였다. 오디오 신호를 DSP 보드에 입력시키면 DSP 프로그램은 이 신호를 실시간으로 처리하여 압축된 MPEG-1 오디오 비트열을 출력시킨다. 출력된 비트열을 PC로 입력받아 복호 해 봄으로써 부호화기가 실시간으로 동작함을 확인하였다. PC 상에서의 복호는 Xing사의 Xing-sound player 프로그램을 사용하였다.

복호 된 오디오 신호의 음질을 테스트하기 위하여 다양한 음악을 선택하여 부호화하였다. 이때의 표본은 팝송과 클래식, 그리고 아프리카 전통 타악기 음악 등으로 나누어 10여 곡을 선정하였다. 그리고 본교 전자공학과 학부 학생 40명을 대상으로 Mean Opinion Score (MOS) 테스트를 수행하였다. 수행 결과, 비록 전문가에 의한 평가는 아니었지만, 전체 테스트의 평균치는 5점 만점에 4.0에 거의 근접한 우수한 수준으로 나타났다. 이를 토대로, 구현된 부호화기의 성능이 일반 불특정 대중을 대상으로 생각할 때, 양호한 것으로 자체 평가하였다.

V. 결 론

본 논문에서는 DSP 단일 칩을 사용하여 실시간 MPEG-1 오디오 부호화 시스템을 구현하였다. 구현된 부호화기는 계층 II를 사용하였고 심리음향모델은 모델-1을 사용하였다. 우선 MPEG-1 오디오 표준을 분석하여 C-언

어로 프로그램 하였다. C-언어 프로그램을 통하여 동작을 이해하고 확인한 후, TI 어셈블리어로 전체 프로그램을 새로 코딩하여 동작 시간의 단축을 꾀하였다. 실제 구현 시 MPEG-1 오디오는 연산량이 매우 많기 때문에 C-언어 프로그램 상에서도 많은 최적화 과정이 필요했으며, 어셈블리어 프로그램 시에도 DSP 칩이 가진 여러 가지 고속 동작 처리를 위한 특성을 충분히 살려야 했었다.

구현한 MPEG-1 오디오 부호화 프로그램을 동작시키기 위하여 DSP 보드를 자체 제작하였는데, 구현된 DSP 보드의 내용은 다음과 같다. 입력 단은 디지털 오디오 전용 ADC인 CS4231A를 사용하였고, DSP CPU로는 실수 연산 DSP 칩인 50 MHz TMS320C31을 사용하였고, 사용된 메모리는 128 kword이다. DSP 보드에서 다른 많은 모듈들은 VHDL로 프로그램하여 Altera 사의 FPGA EPF81188AGC 232-4로 구현하였다.

현재 부호화기의 대상은 48 kHz 샘플링 주파수에 16 비트 스테레오 오디오이고 출력 비트율이 192 kbps로 고정한 최대 조건이므로, 그 이하의 조건에 대하여는 쉽게 변경이 가능하다. 즉 외부로부터의 입력 방법을 모색하여 주파수와 샘플링 주파수, 그리고 출력 비트 속도를 설정해주면 이에 맞추어 부호화 프로그램을 수행하도록 쉽게 변경할 수 있으며, 이를 통하여 MPEG-1 오디오 부호화기의 모든 규격을 만족하는 통합 보드로 확장할 수 있다.

또한 DSP 보드에서 실수 연산 이외의 부분을 VHDL로 구현하면 DSP의 연산 부담을 줄일 수 있었다. 예를 들어, 서브밴드의 데이터를 부호화하는데 필요한 비트 수를 출력시키고 각 서브밴드의 데이터를 출력시켜 주면 FPGA에서 이 정보를 받아서 비트열을 완성시킬 수 있었다. 이와 같이 VHDL을 통하여 계산 부담을 더는 동시에 DSP 칩을 몇 개 더 사용하여 다채널을 처리할 수 있도록 구현하면 MPEG-2 오디오 부호화기로도 확장할 수 있겠다.

참 고 문 헌

[1] 강성훈, "디지탈 오디오와 심리음향", *대한전자공학회지, 제12권 제5호*, pp. 19-29, 1995년 5월.

[2] A. Houtsma, "Psychoacoustics and Modern Digital Audio," *Philips J. Res. 47*, pp. 3-14, 1992.

[3] ISO/IEC JTC1/SC29/WG11 no. 603, *Generic Coding of Moving Pictures and Associated Audio-CD 13818-3 (Part3. MPEG-Audio)*, Nov. 1993.

[4] ISO/IEC JTC1/SC29/WG11 no. 803, *Generic Coding of Moving Pictures and Associated Audio-Audio*, Nov. 1994.

[5] W. Doh, et al., "Audio Coding Algorithm of MPEG and Dolby Lab's AC-3," *제1회 전기음향워크샵*, pp 94-98, 한국전자통신연구소, Oct. 1995.

[6] *TMS320C3x User's Guide*, Texas Instruments, 1993.

[7] D. Perry, *VHDL*, 2nd Ed., Donnelley & Sons Company, 1993.

[8] J. Amstrong and F. *Gray, Structured Logic Design with VHDL*, Prentice-Hall, 1993.

[9] 박현철, *VHDL: 회로설계와 응용*, 한성출판사, 1994.

[10] *Synopsys VHDL Compiler Reference*, Ver. 3.2a.

[11] *Synopsys VSS Expert Core Program Manual*, Ver. 3.2a.

[12] *MAX-PLUS2 Users Guide*, Ver. 3.0, 1992.

저 자 소 개

전 기 용

1972년 8월 18일생
1995년 2월 한양대학교 전자공학과 졸업 (공학사)
1997년 2월 한양대학교 대학원 전자공학과 졸업 (공학석사)
1997년 3월~현재 LG 정보통신 중앙연구소 전송연구단 연구원
주관심분야: 디지털통신, 신호처리, MPEG 시스템

이 동 호

1962년 3월 2일생
1986년 2월 한양대학교 전자공학과 졸업 (공학사)
1988년 12월 University of Texas at Austin 전기 및 컴퓨터공학과 졸업 (공학석사)
1991년 5월 University of Texas at Austin 전기 및 컴퓨터공학과 졸업 (공학박사)
1991년 6월~1994년 2월 LG 전자 중앙연구소 선임연구원
1992년 9월~현재 한양대학교 제어계측공학과 조교수
주관심분야: 영상처리, 영상압축, 디지털 시스템 설계

조 성 호

1960년 2월 21일 생
1982년 2월 한양대학교 전자공학과 졸업 (공학사)
1984년 12월 University of Iowa 전기 및 컴퓨터공학과 졸업 (공학석사)
1989년 8월 University of Utah 전기 및 컴퓨터공학과 졸업 (공학박사)
1989년 8월~1992년 8월 한국전자통신연구소 부호기술부 선임연구원
1992년 9월~현재 한양대학교 전자공학과 조교수
주관심분야: 디지털통신, 무선통신, 신호처리, 정보통신 시스템