특집/오디오 분야

디지털 오디오 복호화 칩의 구현에 관한 연구

차 형 태
숭실대학교 전자공학과

요 약

본 논고에서는 VHDL ASIC 설계 기술을 사용하여 Chip을 설계할 때에 필요한 사항과 방법 그리고 실제 사용 예로써 MPEG 오디오 Chip의 설계와 구현에 관하여 기술한 것이다. VHDL을 이용한 설계의 흐름도로부터 실제 설계를 위한 방법까지 기술하였고 알고리듬의 최적화를 위한 방법과 그 예를 보이고 있다. 또 Gate를 이용한 Logic Level설계에 익숙하지 않은 설계자도 쉽고 빠르게 사용할 수 있는 VHDL 설계 기술을 이용하여 MPEG-2 의 2 채널 모드까지 지원하는 Chip의 설계에 관하여 기술한다. 특히 합성 필터를 설계할 때 계산량을 줄이고 RAM의 크기를 줄일 수 있도록 효율적인 구현을 위해 구조를 설계하였으며 ROM에 저장될 합성 필터 계수의 수를 줄이기 위해 노력하였다. 또 합성 필터의 Control을 위하여 Pseudo_RISC개념을 사용하였다.

Ⅰ. 서 론

대용량 디지털 정보의 처리와 전송을 가능하게 한 기술의 발전은 급속하게 이전의 아날로그 기술을 대체해 나가고 있고 이에 따라 영상과 음성 그리고 데이터를 저장, 압축, 전송, 분석하는 멀티미디어 기술이 주목받고 있다. 하나의 예로써 저장 매체는 DVD가, 전송기법에는 ATM방식, 분석에는 많은 자료검색 기법이 소개되고 있으며 이 방식의 사용을 위한 많은 프로그램들이 개발 중에 있다. 최근에 결정된 MPEG(Moving Picture Export Group) 오디오, 비디오 규격은 압축과 신장에 관한 것으로 디지털 방송에 적합하도록 만들어졌으며 DBS, HDTV, DAB, Multimedia기기 등에 적용이 될 예정이다. 최대 1.5 Mbps이하의 비디오 전송률과 2 채널, 최대 384 Kbps이하의 오디오 전송률을 가진 MPEG-1규격[1]과, 1.5 Mbps이상의 비디오 전송률과 5.1 채널, 최대 약 1066 Kbps 이하의 오디오 전송률을 가진 MPEG-2규격[2]은 각각 1992년과 1994년에 MPEG 회의에서 압축과 신장 그리고 전송을 위한 표준 방식이 결정되었다. 이에 따라 세계 각국에서는 이 규격을 이용하여 디지털 방송을 하기 위한 기초 기술을 습득하기 위해 노력하고 있다. 한국에서도 DBS 시험 방송이 1996년에 시작되었고 수신기 개발을 위해 Chip의 제작을 위한 자체 기술 개발에 노력하고 있다. Chip을 설계할 때에는 직접 gate를 이용하여 설계하는 것이[3][4] 보편적이었으나 요즘 회로의 복잡도가 증가함에 따라 Chip의 설계가 더욱 어려워지고 또 기술의 발전 속도가 가속화됨에 따라 빠른 시간 안에 완성하는 것을 요구함으로 새로운 디자인 도구의 필요성이 높아졌다. 이러한 도구 중의 하나인 VHDL(Very High Speed Integrated Circuit Hardware Description Language)[5][6]은 대용량의 Chip을 설계하는데 적합하도록 만들어진 ASIC 도구이다. VHDL은 배우기가 쉽고 Chip을 설계할 때 필수적으로 요구되는 많은 Gate들에 관한 지식을 필요로 하지 않고 최소한의 지식만으로도 설계가 가능하여 최근에는 많이 사용되고 있다[7].

본 논고에서는 VHDL을 이용한 칩의 구현 방법과 알고리듬의 최적화를 위한 방법에 대하여 설명하고 압축과 복원 알고리듬 중에서 MPEG 오디오 복화화기를 예로 들어 설계기법을 소개한다.

Ⅱ. VHDL 설계

어떤 특정한 기능을 수행하도록 하기 위하여 시스템의 설계가 끝나면 Logic으로 구현하거나 DSP Chip을 이용하거나 또는 ASIC(Application Specific IC) Chip을 설계하여 사용하는 방법이 있다. 이 중 Logic으로 구현하는 방법은 간단한 회로의 설계에 주로 사용되고, DSP Chip은 빠른 구현과 간편하게 수정할 수 있다는 장점이 있으나 Chip이 크고 빠른 속도의 연산 처리가 어렵다는 문제가 있다. 이에 비해 ASIC은 소형화가 가

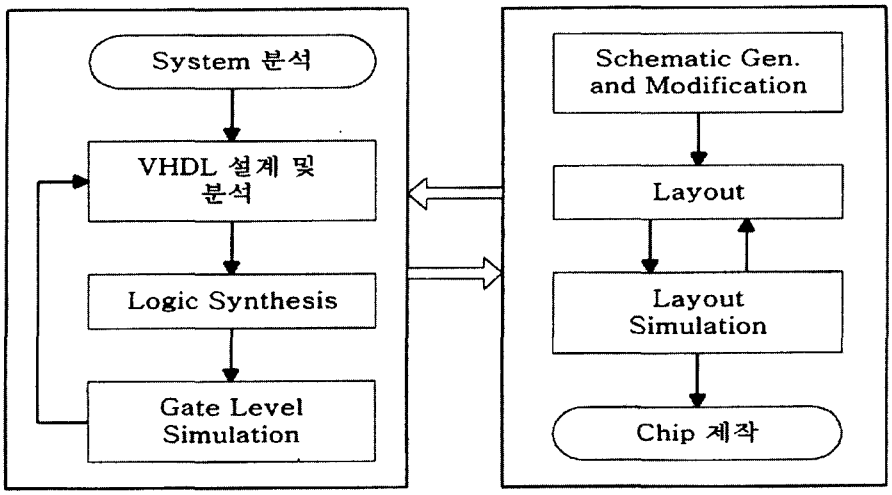


그림 1. VHDL을 이용한 설계 흐름도

능하고 빠르고 많은 계산을 할 수 있다는 장점이 있다. 기술의 발전 속도가 빨라지고 있고 이에 따라 알고리듬의 복잡도가 수십만 Gate 수준으로 증가하면서 Logic으로 구현하거나 DSP Chip 만을 독자적으로 이용하는 것은 한계에 다다르고 있고 빠른 설계에 어려움이 많이 있다.

Logic Level에서의 ASIC Chip 설계방식은 간결하고 효과적인 설계가 가능하다는 장점이 있으나 시간이 많이 걸리고 설계를 위한 기초 지식과 경험이 많이 요구된다는 단점이 있어서 시스템 설계자가 직접 Chip 설계에 참여하기에는 어려운 점이 많이 있었다. 하지만 시스템의 구조와 난이도가 점점 복잡해지고 어려워짐으로써 Chip 설계자와 시스템 설계자가 함께 공동으로 작업할 때 어려운 점이 많이 발생하고 있다. Chip 설계자는 시스템에 대한 지식이 부족하여 그 구조를 이해하기에 많은 시간이 필요하고 시스템 설계자는 Chip 설계에 대한 지식이 거의 없어서 효율적인 알고리듬의 최적화에 어려운 점이 있었다. VHDL을 사용은 이러한 어려운 점을 해소하는데 큰 도움을 준다. 또한 VHDL은 정보의 교환이 용이하고 정확하고 간결한 표현이 가능하고 문서화에 편리하며 광범위한 표현과 확장성을 가지고 있으며 Simulation을 통하여 검증이 용이하고 Top-down방식의 모듈별 설계로 보안이 가능하다는 장점이 있다.

VHDL 설계의 출발은 시스템의 분석으로 부터 시작된다. 시스템의 철저한 분석과 효율화는 Chip의 크기와 성능에 큰 영향을 미치게 된다. 설계 시에 주의할 사항은 모든 VHDL 명령어가 Logic으로 구현되지 않기 때문에 선택적으로 사용하여야 하며 이러한 문제점들은 점차적으로 기술의 발달과 함께 개선되어 가고 있다. 또 VHDL을 이용한 설계의 회로도를 보면 compiler가 gate의 수와 배치를 효율적으로 하기 위해 재배치 함으로 보통 매우 복잡하고 Debugging이 어려울 때가 많다. 따라서 하나의 Module은 너무 크지 않게 설계하는 것이 후에 재 수정할 때 도움이 된다. 그리고 설계 시부터 나중의 Chip검증을 위해 toggling test와 같은 필요한 검사에 대해 대비하는 것이 필요하다.

그림 1.의 과정을 통하여 설계가 끝난 시스템은 Simulation을 통해 검증되며 Chip을 만들 Vendor가 제공하는 Library를 사용하는 Logic Synthesis를 통해 Gate Level로 만들어지고 다시 Simulation을 통해 검증된다. 비록 VHDL로 설계한 회로가 대부분 적절히 Gate Level로 만들어지지만 설계자가 원하지 않는 부분이 만들어지기도 하며 또 비효율적인 구성이 나타나기도 하기 때문에 회로의 수정이 불가피 할 경우가 있다. 이를 위하여 Schematic을 만들어서 직접 회로를 수정을 하는 것이 시간을 절약하는 방법이 될 수 있다. 모든 수정이 끝나면 Layout을 만든 후에 다시 검증하고 Chip제작을 하게 된다.

이 설계 흐름도에서 시스템 설계자에게 가장 중요한 부분은 설계의 최적화 단계이다. 설계의 최적화를 위해서는 알고리듬의 철저한 분석이 필요하고 또 Chip설계를 위한 기초단계로써 Clock Diagram을 그려볼 필요가 있다. 다음 (1) 식의 계산을 수행하는 Chip을 구현하기 위한 최적화 시의 고려사항들을 살펴보면,

$$z[k] = (x[N/2 - 2k - 1] + jx[2k]) \times -e^{2\pi(8k+1)/(8N)} \quad (1)$$

이때 N=512이고 k=0..127이다.

주어진 Clock의 속도에 맞추어 4개의 값으로 표현된 한 블록에 하나의 복소수 계산을 하기 위하여 입력 RAM으로부터 두 개의 값을 읽은 후에 ROM으로부터 두 개의 계수값을 읽어서 주어진 입력값과 곱하게 된다. 4번의 곱하기가 이루어지는 동안에 한번의 더하기와 빼기를 수행하고 그 값을 다시 RAM에 저장한다. 이때 RAM과 ROM의 주소를 만들어 주기 위하여 Counter를 이용하고 이 값의 역수를 사용하여 x[k]의 두 개 값을 읽어 사용하게 된다. 이때 중요한 점은 한 블록에서의 계산을 위하여 각 연산의 순서와 방법을 효율적으로 분배하고 각 Memory의 값들을 정해진 위치에서 사용할 수 있도록 하는 것이다. 그림 2.에 이 방법이 보이고 있다.

Ⅲ. MPEG 오디오 Chip의 구현

MPEG 오디오 부호화기에서 만들어진 비트열은 그림 3.에서와 같이 크게 4가지 부분으로 나눌 수 있으며 다음과 같다.

첫째, Header부분은 동기신호, 비트열의 압축률, 압축방법, Sampling 주파수, Stereo 또는 Mono 방송의 여부 등에 대한 기본적인 정보를 가지고 있으며Error에 가장 민감한 부분이다. 둘째, CRC (Cyclic Redundancy Code)는 비트열의 주요 부분에 Error의 발생 여부를 검사하는 부분이며 동기신호를 제외한 Header부분에서부터 Scfsi까지 주어진 16bit 방정식에 의하여 계산된다. 셋째로 Audio Data부분은 실제 부호화된 오디오 신호에 대한 정보를 가지고 있으며 Bit 할당량(Bit Alloc.), 크기정

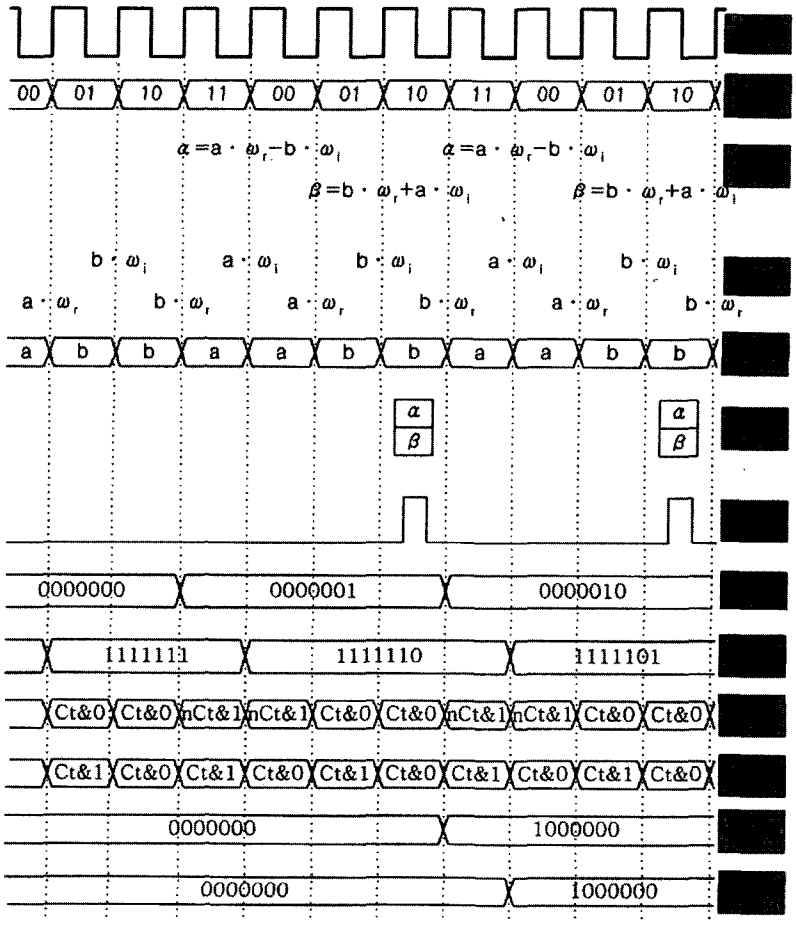


그림 2. Chip설계를 위한 Clock Diagram

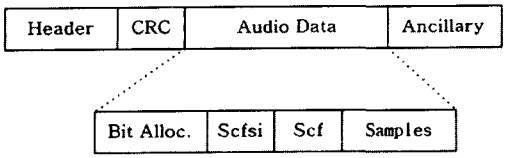


그림 3. MPEG 비트열의 구조

보 선택(scfsi), 크기정보(scf) 그리고 Sample들로 이루어져 있다. 이 정보들을 바탕으로 역양자화와 합성 필터를 거쳐 PCM신호로 복호화된다. 마지막으로 Ancillary 부는 부가정보를 전송하고자 할 때 사용되는 부분이다.

이 비트열이 복호화기에 입력되고 동기신호가 검출되면 Parser부에서 각 부분별로 정보들을 분류하여 저장한다. 압축된 Sample들까지 복호화가 끝나면 역양자화기에서 역양자화되고 크기정보를 이용하여 원래의 크기로 복원된다. 이 신호들은 합성필터부에서 합성되고 PCM신호로 재생되어 출력하게 된다.

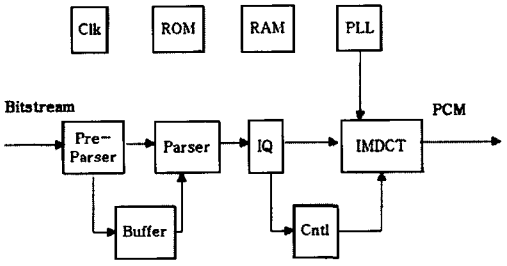


그림 4. MPEG복호화기의 구성도

Frame단위로 압축된 신호를 포함하고 있는 비트열이 수신기에 들어오면 그림 4.에서와 같이 Pre-Parser에서 동기신호를 검출하게 되고 그 후에 Buffer에 저장하게 된다. 이 Buffer는 MPEG-2 비트열이 입력될 경우를 생각해서 그 길이를 결정하게 되며 본 연구에서는 384 Bytes를 사용하였다. 이 Buffer는 Dual Port RAM을 사용하여 신호의 입출력이 자유롭게 설계되었으며 항상 일정한 양의 Data를 유지하도록 하였다. 이Data는 Parser에서 필요에 따라 출력하여 사용하게 되며 비트열에 포함된 정보들을 해독하여 다음 단에 보내게 된다. 역양자화되고 본래의 크기로 회복된 Data는 합성필터부에서 합성되어 PCM신호로 전환된다. 이때 역양자화 계수와 합성필터 계수는 ROM에 저장되어 있어 필요에 따라 사용되며 이 모든 과정은 Control부에서 통제된다. 모든 부분에서 사용되는 Clock은 주 주파수인 27MHz에서 분주 되어 사용되며 Clock부에서 만들어지

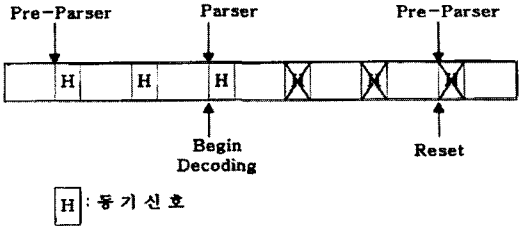


그림 5. 동기신호의 검출

게 된다. PCM신호로 재생된 오디오 신호는 48 kHz, 44.1 kHz 그리고 32 kHz로 출력되며 이때 27 MHz로 부호화된 비트열과의 동기를 유지하기 위하여 PLL(Phase Locked Loop)를 사용한다.

MPEG 오디오 비트열에는 각 Frame의 시작을 표시하는 동기 신호가 들어있으며 12개의 1로 구성되어있다. 그러나 이 신호는 Unique하지 않기 때문에 Frame의 중간에 나타날 수도 있으며 이것은 각 Frame들이 잘못 동기 되게 하여 복호화를 방해하는 요인으로 작용할 수 있다. 그러므로 동기신호의 정확한 검출은 매우 중요하다고 할 수 있다. 동기신호는 Sampling 주파수가 고정되었을 때 각 Frame간의 간격이 일정하다는 사실을 이용하여 검출될 수 있다. 따라서 그림 5. 에서와 같이 일정한 간격으로 3번 이상 동기신호가 검출되었을 때에만 진정한 동기신호로 인정하여 복호화를 진행하게 된다. 복호화 진행 중에 동기신호의 에러가 발생하면 2번까지 에러 보정을 한 후에 다시 처음으로 돌아가 동기신호의 검출부터 새로 시작하게 된다.

이때에 계층-2에 사용되는 한 Frame의 길이는 식 (2)과 같이 계산할 수 있다.

$$Bytes = 144 \times Bitrate / Sampling\_Frequency \quad (2)$$

예를 들어 48kHz의 주파수와 384kbps의 압축률을 사용하면 1 Frame은1152개의 Byte로 구성되어 있으며 이때 주파수와 압축률은 각 Frame의 Header부분에 표시되어 있으며 이를 이용하여 각 Frame의 길이를 구할 수 있다.

동기신호가 확인된 후에 그림 6. 에서와 같이 Parser(P)에 비트열이 입력되면 Parser가 동작되고 역양자화기(IQ)를 거쳐서 합성필터(IMDCT)에서 PCM신호로 재생된다. 이때 초기화가 진행되는 처음 부분을 제외하고는 Parser 부가 합성필터부와 동시에 동작하게 되며 이것은 빠른 신호의 복호화를 가능케 한다. 두 번째 IQ가 시작될 때 합성필터를 통해 만들어진 PCM 신호들이 출력되기 시작하며 이때 다음 번 PCM신호들이 처리된다. 두 Parser 사이의 길이는 1 Granule, 즉 32개씩

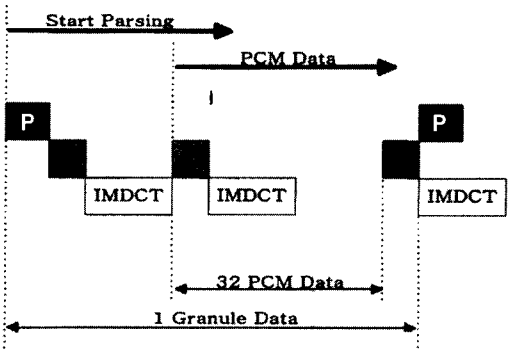


그림 6. 각 Block의 Timing 분석

3개로 이루어진 96개의 PCM신호이며 두 IQ 사이의 길이는 32개의 PCM신호가 출력되는 시간과 같다. 이 복호화기가 48kHz의 Sampling주파수를 사용하면 1개의 PCM신호가 출력될 때 필요한 시간은 약 20.83s이며 32 PCM신호를 위해서는 약 667s의 시간이 요구된다. 따라서 Parser는 이 시간 안에 신호 분석을 끝내면 되고 IQ와 IMDCT를 계산하기 위하여 필요한 시간은 667s 보다 짧으면 된다.

만일 주 동작 주파수를 분주하여 각 부분에 필요한 Clock으로 사용한다면 이것을 이용해서 각 부분에 필요한 Clock의 빠르기를 결정할 수 있다. 예를 들어 Parser부는 처리해야 할 신호가 IMDCT부에 비하여 아주 적기 때문에 IMDCT를 처리하는 속도로 Parser부를 처리하면 전력의 소모가 많아지므로 낮은 주파수를 사용하는 것이 효과적이다. 이때 다른 주파수를 사용함으로써 발생할 수 있는 혼돈은 Parser부와 IMDCT부가 서로 독립적으로 동작하게 함으로써 피할 수 있다. 또한 두개의 다른 주파수라도 하나의 주된 주파수에서 분주 되어 만들어짐으로 해서 동기 되어 있기 때문에 문제가 없다.

시스템을 구성하기 위해서는 각 부분의 특성과 동작을 이해할 필요가 있다. Parser부에서는 동기 검출된 비트열로 부터 Header정보, Bit allocation, Scale factor information, Scale factor select information그리고 Sample Data들을 분석하고 저장한다. Header정보는 비트열의 특성에 관한 자세한 기록, 즉, Sampling 주파수, Bitrate, Stereomode등등을 가지고 있다. Sample Data는 Bit의 양에 따라 Grouping을 해서 보내는 경우가 있는데 이것은 다음과 같이 Ungrouping할 수 있다.

```
for i=0:2
 sample[i] := c%nlevels;
 c := c/nlevels;
end
```

여기에서 c는 Grouping된 Sample이며 nlevels는 Grouping된 3개 Sample의 Bit수이다.

역양자화 부에서는 복호화 된 Sample Data에 역양자화 계수를 더하고 곱한 후에 Scale factor를 곱하여 구해진다.

S' := C×(sample+D);
S := S'×Scalefactor;

C와 D는 역양자화 계수이며 Scalefactor는 Parser부에서 해석된 크기 조정 계수이다.

$$Y[i] = \sum_{k=0}^{31} \cos\left(\frac{(16+i)(2k+1)\pi}{64}\right) \times S_k \qquad (3)$$

식 (3)의 방정식을 이용하여 32개의 Sk로 부터 64개의 Y[i]를 구하여 사용해야 하지만 계산량을 줄이기 위해 cosine함수의 대칭성을 사용하여 32개의 Y[i]만으로 64개의 Y[i]를 구할 수 있다. 즉,

$$Y[i] = Y[31i],\ Y[i32] = Y[63i] \qquad (4)$$

여기서 i = 0, 1, ... ,15 이다.

이렇게 구하여진 Y[i]는 64개씩 Shifting되는 Vi로 대입된다.

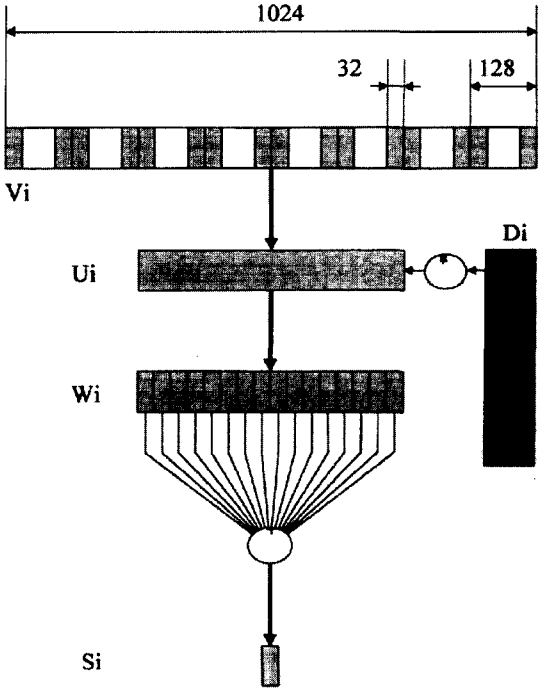


그림 7. MPEG 합성 필터의 구조

$$V[i] = V[i-64],\ i=1023, \ldots, 65,64$$
$$V[i] = Y[i],\quad i=63, \ldots, 2,1,0 \qquad (5)$$

이때 1024개의 Shift는 많은 시간과 복잡한 회로를 요구하며 이는 순환 Queue를 사용하여 해결할 수 있다.

Vi는 그림 7. 에서와 같이 1024개의 Word로 이루어져 있으나 Cosine함수의 대칭성을 이용하여 512개의 Word만 저장하고 이로부터 1024개의 계수들을 구할 수 있다. Ui는 Vi를 8개의 블록으로 만들어 1 블록의 128개에서 앞뒤 32개씩을 취하여 만들어 진다. 이 Ui에 Window계수 Di를 곱하여 512개의 Wi를 만들고, Wi를 32개의 계수를 가진 16개의 블록으로 만들어 각 블록을 더하여 32개의 Si를 만든다.

또 32개의 PCM값인 Si를 구할 때 Ui와 Wi와 같은 RAM을 거치지 않고 직접 계산함으로써 불필요한 RAM부분을 줄일 수 있다. Vi부터 Si까지는 한 경로를 사용하여 계산함으로써 설계의 효율성을 기할 수 있고, 불필요한 출력 부분의 시간 지연을 최소화 할 수 있다.

표 1. 합성필터 처리를 위한 명령어

	Control	Jump	Add ?	Mul ?
0	CREPT	1	NO	NO
1	MORML	2	IQ_D	NO
2	NORML	3	NO	IQ_C
3	CLOOP	1	NO	SCALE
4	DELAY	4	NO	IQDLY
5	NORML	6	NO	IMDCT
6	CREPT	7	IMDCT	IMDCT
7	NORML	8	IMDCT	NO
8	NORML	9	NO	NO
9	NORML	10	NO	WINDOW
10	CREPT	11	WINDOW	WINDOW
11	CLOOP	4	WINDOW	NO
12	CLOOP	4	NO	NO
13	CJMPR	1	NO	NO
14	CJMPR	1	NO	NO
15	CRET	1	NO	NO

$$S_i = \sum_{k=0}^{7} [V_{128k+i} D_{64k+1} + V_{128k+i+96} D_{64k+i+32}] \qquad (6)$$

이때 Vi와 Si의 관계는 주어진 식 (6)으로 부터 직접 구할 수 있다.

IMDCT 계수는 사용되는 cosine함수가 128개의 주기를 가지고 있음으로 대칭성을 이용 약간의 주변 회로를 추가함으로써ROM에 저장해야 하는 2048개 계수의 수를 128개, 64개 또는 32개로 줄일 수 있으며 이때 계수의 수가 줄어들 수록 주변 회로의 복잡도가 늘어나게 된다. 이러한 방법들은 설계자에게 다양한 선택을 할 수 있게 도와주며 경우에 따라 사용 방법을 다르게 하여 효율적인 설계를 할 수 있다.

```
data := 0~31
step := data<<1;
idx := data;
for i=0:31
  idx := idx % 128;
  coeff(i) := cos_array(idx);
  idx := idx + step;
end
```

위에서 보이는 바와 같이 index가 초기치인 data와 변화의 크기인 step을 가지고 128개의 cos_array에서 원하는 계수 coeff를 구하는 방법을 기술하고 있다. 초기치인 data는 0에서부터 31까지 변화하며 step은 data값의 두 배이다. 우리가 구하고자 하는 계수의 index값인 idx는 128로 나누었을 때의 나머지이며 이 idx값은 31번 step값 만큼 증가한다. 이 idx에 해당하는 cos_array값이 우리가 원하는 cosine 계수 값이다.

역양자화기와 합성필터를 동작시키기 위해서는 Control이 간단하며 쉽게 수정할 수 있는 장점이 있는 Pseudo-RISC 개념을 사용하였다. Parser에서 분리된 Sample들은 역양자화기를 거쳐 합성필터에서 PCM신호로 만들어지며 이때 Controller에서 정의된 순서에 의해 처리된다. 먼저 필요한 6개의 명령어들을 정의하였고 표 1.에서 보이는 바와 같이 그 기능들이 설정되었다.

명령어 NORML은 무조건 다음 번지로 가라는 명령이며 CREPT는 주어진 조건이 만족될 때까지 그 번지

표 2. Pseudo-RISC 명령어

Command	Operator
NORML	Go to the next address
CREPT	Repeat
CLOOP	Loop
CJMPR	Jump to the specific address
CRET	Go to the specific address
DELAY	Delay

의 명령을 반복적으로 수행하고 다음에 지정된 번지로 jump하라는 명령어이다. CLOOP는 지정된 번지로 jump 하여 명령을 수행하고 조건이 만족되면 다음 번지로 가라는 명령어이다. CJMPR는 단순히 지정된 번지로 jump하라는 명령어이며 CRET는 처음부터 새로 시작하라는 명령어이다. DELAY는 그 번지에서 지정된 시간만큼 머문 후에 다음 번지로 가라는 명령어이다.

이 명령어들을 이용하여 표 2에서 보이는 바와 같이 16개의 순서대로 순차적으로 처리를 하게 되며 그 기능들은 다음과 같다. 기본적인 동작은 한 주소의 명령이 완전하게 이루어 졌으면 Control에 사용된 명령어에 따라 다음 주소나 Jump주소로 가고 그렇지 않으면 그 주소의 명령을 계속 수행하도록 되어있다. 주소 0 ~ 3번은 역양자화기에 관한 것이며 역양자화 계수 D(IQ_D)를 더하고 C(IQ_C)를 곱한 후에 Scalefactor (SCALE)를 곱하는 것으로 이루어진다. 주소 4 ~ 8번에서는 IMDCT계수를 곱하고 더하는 과정이며 주소 9 ~ 11번에서는 Window계수를 곱하고 더하게 된다. 이러한 과정이 한 Frame이 끝날 때까지 지속되며 주소 12 ~ 15번까지의 명령어를 이용하여 이 과정을 통제하게 된다.

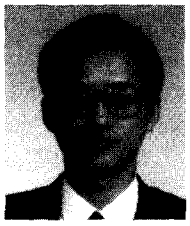
Ⅳ. 결 론

본 논고에서는 VHDL ASIC 디자인 도구를 이용하는 방법과 그 장단점에 대하여 기술하고 예로써 MPEG 오디오 복호화기를 구현하였다. VHDL의 사용은 정보의 교환을 용이하게 하고 정확하고 간결한 표현과 문서화에 편리하며 광범위한 표현과 확장성을 가지고 있고 Simulation을 통하여 검증이 용이하며 모듈별 설계로 보안이 가능하다는 장점이 있다. VHDL을 이용한 설계의 예로써 MPEG-2의 2채널까지 지원하는 이 복호화기를 설계하기 위하여 System의 Clock Timing을 분석하였고 27MHz의 주 Clock에 맞추어 사용할 수 있도록 각 블록별로 Clock Timing 을 조정하였다. 또 동기신호 검출의 분석을 통하여 error가 발생하였을 때 비트열의 동기를 유지하도록 하였다. 특히 합성 필터부의 RAM과 ROM의 크기 그리고 각 부분의 연산에 필요한 계산량을 최적화 하기 위하여 알고리듬을 효율적으로 적용하였으며 이 부분의 Control을 위하여 Pseudo-RISC를 설계하여 사용하였다.

참 고 문 헌

1. ISO/IEC 11172-3 Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to 1.5 Mbit/s - Audio Part, International Standard, Nov. 1992
2. ISO/IEC 13818-3 Generic Coding of Moving Pictures and Associated Audio - Audio Part, International Standard, Nov. 1994
3. Greg Maturi, Single Chip MPEG Audio Decoder, IEEE Transaction on Consumer Electronics, Vol. 38, No. 3, Aug. 1992
4. Tsung-Han Tsai and et al., An MPEG Audio Decoder Chip, IEEE Trans. on Consumer Electronics, Vol. 41, No. 1, Feb. 1995
5. Jean-Michel Berge and et al., VHDL92, Kluwer Academic Publishers, Boston, 1992
6. 최기영, VHDL의 이해, 출판도서 기한재, 1월, 1995
7. 차형태, MPEG 오디오 합성 필터의 구현을 위한 연구, 제 13회 음성통신 및 신호처리 워크샵 논문집, pp79-84, 8월16일, 1996
8. 차형태, VHDL을 이용한 MPEG 오디오 칩 구현을 위한 연구, 한국음향학회지, pp47-52, 제16권 제7호, 10, 1997

필자소개



차 형 태

- *1985년 숭실대학교 전자공학과 졸업(공학사)*
- *1988년 Univ of Pittsburgh Electrical Engineering 졸업(공학석사)*
- *1993년 Univ of Pittsburgh Electrical Engineering 졸업(공학박사)*
- *1993년 ~1996년 삼성전자 신호처리 연구소 선임 연구원*
- *1996년 ~ 현재 숭실대학교 전자 공학과 교수*
- *주관심 분야 : 영상 및 오디오 신호처리, ASIC 칩 설계*