Motion JPEG2000을 위한 실시간 비디오 압축 프로세서의 하드웨어 구조 및 설계

論 文
53D-1-1

Hardware Architecture and its Design of Real-Time Video Compression Processor for Motion JPEG2000

徐 英 鎬* · 金 東 郁**
(Young-Ho Seo · Dong-Wook Kim)

Abstract - In this paper, we proposed a hardware(H/W) structure which can compress and recontruct the input image in real time operation and implemented it into a FPGA platform using VHDL(VHSIC Hardware Description Language). All the image processing element to process both compression and reconstruction in a FPGA were considered each of them was mapped into a H/W with the efficient structure for FPGA. We used the DWT(discrete wavelet transform) which transforms the data from spatial domain to the frequency domain, because use considered the motion JPEG2000 as the application. The implemented H/W is separated to both the data path part and the control part. The data path part consisted of the image processing blocks and the data processing blocks. The image processing blocks consisted of the *DWT Kernel* for the filtering by DWT, *Quantizer/Huffman Encoder*, *Inverse Adder/Buffer* for adding the low frequency coefficient to the high frequency one in the inverse DWT operation, and *Huffman Decoder*. Also there existed the interface blocks for communicating with the external application environments and the timing blocks for buffering between the internal blocks. The global operations of the designed H/W are the image compression and the reconstruction, and it is operated by the unit of a field synchronized with the A/D converter. The implemented H/W used the 54%(12943) LAB(Logic Array Block) and 9%(28352) ESB(Embedded System Block) in the APEX20KC EP20K600CB652-7 FPGA chip of ALTERA, and stably operated in the 70MHz clock frequency. So we verified the real time operation, that is, processing 60 fields/sec(30 frames/sec).

Key Words : Hardware, Motion JPEG, Vedio Compression, DWT, Wavelet

1. 서 론

대용량의 데이터를 갖는 영상 및 비디오를 한정된 채널 대역폭을 통해 이동시키고 제한된 저장매체에서 효율적으로 저장하기 위해 다양한 형태의 압축 기법이 연구되어 왔는데, 그 대표적인 기법이 JPEG, MPEG과 H.26X 관련 표준들이다. 이러한 표준들은 DCT(Discrete Cosine Transform)를 기반으로 하고 있고 지금까지 연구소 및 산업체에서 많은 연구가 이루어져왔다. 그러나 정보량의 폭발적인 증가와 서비스 형태의 다양화, 그리고 무선통신에 기반한 모바일 서비스의 확대는 더 향상된 데이터 압축기술의 개발과 새로운 방식을 요구하고 있다. 이러한 요구에 대해 가장 근접하는 차세대 기술로서 이산 웨이블릿 변환(Discrete Wavelet Transform, DWT)이 저변을 확대해 나가고 있는데, 이미 DWT는 JPEG2000표준에서 주 변환기법으로 채택된 주파수 변환기법으로 DCT와 달리 블록효과가 발생하지 않고 전체영상을 대상으로 인간의 시각에 따른 처리가 용이하며 주파수 변환 후에 나타나는 다해상도 특성에 대한 다양한 응용이 가능하다. 이러한 특성으로 인해 DCT와 비교할 때 고압축율에서 훨씬 우수한 영상복원 능력을 보이고 있다.

DWT를 하드웨어로 구현하는 연구는 Knowles[1]로부터 시작되었는데, 이 연구는 1차원 DWT를 타겟으로 하고 있으며 중간 연산결과를 다음 연산에 사용하기 위해 다수 또는 대형 다중화기를 사용하여 실제적인 H/W 구현으로는 적합하지 않았다. 이 연구는 Lewis[2]의 연구로 이어져 DWT를 수행하는데 필요한 곱셈기의 사용을 피하는 알고리즘을 제안하였으나, 너무 제한적인 DWT 필터에만 적용 가능하여 일반적인 영상들에 대한 범용성이 매우 약하다. 1차원 DWT를 수행하는데 있어서 메모리를 효과적으로 사용하고자 하는 연구도 진행되었다[3].

2차원 DWT에 대한 연구 중 가장 많은 비중을 차지하는 부분이 DWT를 수행하는 순서나 DWT 연산방법을 변경하고자 하는 것이다. 먼저, 1차원 DWT를 2차원으로 확장하여 분리 가능한(separable) 2차원 DWT 방법과 행과 열의 변환이 분리 불가능(non-separable)한 방법으로 나누어 볼 수 있다. 분리 불가능한 알고리즘은 복잡성 때문에 많은 레지스터를 사용하여야 하고 스케줄링이 복잡하여 근본적으로 분리 가능한 방법들에 비해 많은 재원을 필요로 한다. 분리 가능한 방법 중에는 영상의 주기적인 확장을 통하여 배치(batch) 처리가 가능하도록 하는 연산방식[4]과, 저대역 통과 및 고대역 통과과정을 병렬로 수행하는 방법[5], 연산 스케줄 표를 형성하여 연산순서를 결정하도록 하는 방법[6] 등이 제안되

* 正 會 員 : 光云大學校 電子材料工學科 博士課程
** 正 會 員 : 光云大學校 電子材料工學科 教授 · 工博
接受日字 : 2003年 7月 15日
最終完了 : 2003年 11月 9日

었다. 특히 Vishiwanath[7]는 systolic 어레이와 파이프라인 구조를 결합한 연산형태를 제안하였다. 이 외에 DWT를 효과적으로 수행하도록 하는 노력은 필터뱅크를 형성하고 필요에 따라 필터의 종류 및 필터의 길이를 조절할 수 있도록 한 방법[8][9] 등 다방면에서 연구가 진행되고 있으며, DWT를 수행하는 H/W를 IP화 하고자 하는 노력[10]과 FPGA에 사상하는 방법[11]도 연구되고 있다.

최근 들어서는 리프팅(lifting) 기법을 이용한 H/W구현 연구가 많이 이루어지고 있는데, Jiang[12]은 리프팅 기법을 이용한 영상의 분할을 통해 메모리 사용량과 접근 횟수를 줄이는 방법을 제안하였다. Robert[13]은 (5,3) 리프팅 필터에 interleaving을 도입하였고 Bolsens[14]는 리프팅 기반의 제로트리 알고리즘을 적용하고 새로운 interleaving 기법을 제안하였다. 또한 Chakrabarti[15]는 JPEG2000을 위한 리프팅 방식의 H/W를 제안하였다.

리프팅 기법이 컨벌루션(convolution)에 기반한 필터링 기법에 비해서 소프트웨어적인 메모리량의 감소와 메모리에 대한 접근 횟수의 감소를 가져오고 웨이블릿 변환과 역변환이 동일한 구조로 이루어진다는 장점을 가지고 있다. 그러나 웨이블릿 필터가 가지는 비인과성(noncausality)에 인과성(causality)을 부여하는 과정에서 리프팅 기법은 필터링 방식에 비해 부가적인 지연이 발생하고 동기화가 복잡하며 이를 위해 ad-hoc FIFO(First Input First Output)가 요구되는 단점을 가진다. 그리고 주 연산을 수행하는 커널(kernel)의 확장성과 프로그래머블한(programmable) 특성을 부여하는 것도 리프팅 방식이 불리하고 경계처리 역시 리프팅 방식이 필터링 방식에 비해서 복잡하다. 마지막으로 H/W 구현 시 외부 메모리로의 접근 횟수는 리프팅 기법이 작지만 내부 메모리의 접근 횟수는 많고 데이터 관리가 복잡한 단점을 가진다. 따라서 구현하고자 하는 H/W의 특성에 따라서 리프팅 기법과 필터링 기법을 선택하고 적절하게 구현해야 하는데 Mattaveli[16]는 이러한 것들을 고려하여 H/W의 유연성과 프로그래머블 특성을 위해 필터링 기법을 사용하는 H/W 구조를 제안하였다.

데이터 압축과정은 양자화 과정에서 주로 이루어지며 그 알고리즘에 따라 압축률과 그에 따르는 영상의 화질이 결정된다. 양자화 과정은 주로 스칼라(scalar) 양자화와 벡터(vector) 양자화로 구분되며, H/W 구현에는 주로 스칼라 양자화 방법이 연구되어 왔으나 최근 들어 가장 활발히 연구되고 있는 방법은 부대역간의 제로 트리(zero tree)를 H/W로 구현하는 방법들이다[17][18]. 양자화 방법에 관련된 연구들은 그 대상을 양자화기에 국한시킴으로써 DWT 수행에 의한 H/W 전체의 동작을 충분히 고려하고 있지 않다. 또한 이들 대부분의 방법들은 메모리 참조 및 양자화 과정과 결합된 DWT 방식을 배제하였으므로 실제의 시스템을 설계하는데 있어서는 많은 문제점을 안고 있다. 본 논문에서는 JPEG2000에서 채택한 선형 양자화기를 사용하고 웨이블릿 변환과 양자화기 그리고 엔트로피 코딩이 모두 고려된 H/W를 제안하고자 한다.

본 논문에서는 2차원 DWT(2D DWT)를 이용하는 디지털 영상압축 및 복원 H/W를 작은 H/W 자원을 가지면서 FPGA에서 실시간 동작이 가능하도록 설계하고자 한다. 가능한한 최소의 H/W 자원을 사용하여 구현하고 이를 하나의 FPGA에 사상한다. 필터링을 수행하는 커널부는 확장이 가능한 병렬적인 구조를 가져 커널을 구성하는 셀(cell) 추가에 따라서 선형적인 데이터 처리량의 증가를 가질수 있도록 한다. 또한 실시간 동작이 가능하도록 효율적인 H/W 구조와 데이터 흐름을 제안한다.

본 논문의 II장에서는 전체적인 H/W의 구조와 규격에 대해서 설명하고 III장에서는 II장에서 언급된 H/W 블록들에 대한 세부적인 구조를 살펴본다. IV장에서는 영상 압축을 위해 어떠한 방식으로 H/W가 동작하는지 설명하고 V장에서 구현결과를 보인다. 마지막으로 VI장에 결론과 향후 연구계획을 밝힌다.

2. 영상처리 H/W의 전체적인 구조와 규격

본 장에서는 구현된 H/W의 전체적인 구조를 설명하고 개략적인 규격에 대해서 설명하고자 한다. 그림 1에 전체적인 구조를 나타냈는데 동작적인 역할에 따라서 크게 데이터 패스부와 제어부(그림에서는 사선으로 표시)로 구분되고 데이터 패스부는 영상 처리블록과 데이터처리 블록으로 나누어진다. 영상처리 블록은 그림 1에서 회색으로 표시된 블록들에 해당하는데, 2차원 DWT를 위한 필터링을 수행하는 DWT 커널부(*DWT Kernel*), 양자화 구역의 확률적 분포를 이용하여 구현된 결합된 형태의 양자화기/허프만 인코더(*Quantizer/Huffman Encoder*), 역 DWT 변환 후에 저주파와 고주파 계수를 더하고 시간적인 타이밍을 조절하는 역변환 덧셈기/버퍼(*Inverse Adder/Buffer*), 그리고 역변환과정에서 허프만 코드를 양자화 계수로 복원하는 허프만 디코더(*Huffman Decoder*)로 구성된다. 또한 그림 1에서 앞서 설명한 블록들이외의 블록들은 데이터처리 블록에 해당하고 표준 입출력과 통신하기 위한 입력 인터페이스(*Input Interface*)와 PLX 인터페이스(*PLX Interface*), 외부 메모리와의 통신을 위한 메모리 버퍼(*Memory Buffer*), 커널과 메모리 버퍼사이의 시간적 완충과 데이터 재정렬을 위한 KtM 버퍼(*Kernel-to-Memory Buffer*), 그리고 가변길이의 허프만 코드를 일정한 길이로 만드는 출력 버퍼(*Output Buffer*)들로 구성된다. 표 1에 구현될 H/W에 대한 전체적인 설계 규격을 나타냈다. A/D 변환기에 맞추어서 NTSC 방식의 영상을 필드단위로 처리하게 되는데 이러한 칼라 영상에 대해서 초당 30프레임의 처리를 통한 실시간 동작을 할 때 약 45대 1의 압축률에서 약 30dB의 화질을 가진다. 내부적으로 웨이블릿 계수는 16비트(정수 9비트, 소수 7비트)를 사용하고 필터 계수는 12비트(정수 2비트, 소수 10비트)를 사용한다. Daubechies (9,7) 필터를 Booth 인코딩된 형태로 내장하여 4 레벨의 DWT를 수행하고 주파수 영역으로 변환한 후에 선형 양자화와 허프만 코딩을 이용하여 영상을 압축한다. 아날로그 신호의 처리를 위해 상용 IC(Bt829b, Bt866)를 사용하며 컴퓨터와 PLX9050을 이용한 PCI 통신을 한다.

3. 영상처리 H/W의 내부 구조

본 장에서는 앞장에서 설명한 전체적인 H/W를 구성하는 각각의 블록에 대한 구조와 역할에 대해서 설명한다.

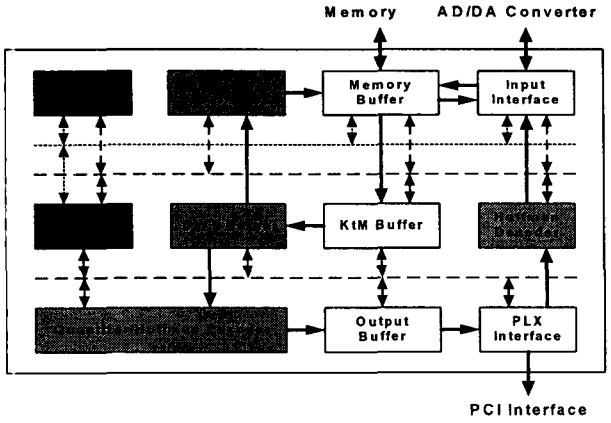

그림 1 하드웨어의 전체적인 구조

Fig. 1 Global architecture of the H/W to be implemented

표 1 하드웨어의 설계 규격

Table 1 Design specification of the H/W

#	category	Specification
1	Image size	640x240(field)
2	Image form	NTSC YCbCr(4:2:2)
3	Compression rate	about 45:1
4	PSNR	about 30dB
5	Number system	Pixel:16-bit(9,7),Filter:12-bit(2,10)
6	Performance	67 field/sec(33frame/sec)
7	DWT level	4 level(octave)
8	DWT filter	Daubechies(9,7) filter
9	Quantizer	Quantisation with Exception
10	Entropy coding	Huffman coding
11	External Memory	512x16bits SDRAM(4)
12	A/D Converter	BT829b
13	D/A Converter	BT866
14	PC interface	32-bit PCI interface(using PLX9050)
15	Target device	APEX20KC EP20K600CB652-7

3.1. DWT 커널의 H/W 구조

DWT 커널은 웨이블릿 변환과 역변환을 위한 컨벌루션 형태의 필터링을 수행하는 블록이다. 메모리에 저장된 필드 단위의 영상데이터는 메모리 버퍼와 KtM 버퍼를 거쳐 DWT 커널로 입력되고 필터링이 수행된다. 레벨단위의 필터링 동작과 함께 최종적인 웨이블릿 계수는 양자화기/허프만 인코더 블록을 통해 압축된 비트열이 된다.

그림 2에 나타난 것과 같이 DWT 커널은 커널 셀(kernel cell)의 병렬적 확장으로 구성되는데, 커널 셀은 FPGA의 ESB를 이용한 내부 듀얼-포트 램(Dual-port RAM)으로 구성된 RAM 체인(*RAM Chain*), 32비트 *CLA*(Carry Look Ahead Adder)로 구성된 선-덧셈(*Pre-Adder*), 그리고 누적 기능을 가진 하이브리드 CSA 트리(*Hybrid Carry Save Adder tree*)와 Booth 곱셈기 및 *CLA*로 구성된 *MAC*(Multiplier and Accumulator, *MAC*) 열의 구조를 갖고 있다[19]. 또한 커널 셀의 입력을 위한 다중 쉬프터(Multi-Shifter)로 이루어진 프리-버퍼(*Pre-buffer*)가 입력단을 구성한다. 본 논문에서 구현하고자 하는 H/W의 내부 연산 블록들은 33MHz의 기준 주파수를 사용하는 것을 가정하는데 이 경우 초당 30 프레임(60 필드)이상의 성능을 보이기 위해서는 1개의 커널 셀로는 불가능하다. 또한 기존 연구에서와 같이 많은 수의 곱셈기와 덧셈기를 사용할 경우 H/W의 양이 과다해질 뿐만 아니라 불필요한 성능(하나의 A/D 변환기에 대해 30 필드 이상의 성능)을 가질 수 도 있는데 A/D 변환기의 출력이 25 프레임에서 30 프레임임을 감안하면 그 이상의 성능은 불필요하고 H/W의 양만을 증가시켜 비용의 손실을 가져온다. 따라서 본 논문에서는 최소의 H/W 자원을 사용하면서 실시간성을 가지도록 4개의 커널 셀을 사용하는 DWT 커널구조로 설계하였다. 다중 채널을 위해 더 높은 필터링 성능이 요구된다면 그림 2에 나타난 것과 같이 병렬적으로 커널 셀을 확장하여 선형적으로 성능의 향상을 가져올 수 있다.

Y, Cb, 그리고 Cr의 컬러 색차성분을 가지는 영상 데이터를 웨이블릿 필터링 할 경우 프리-버퍼는 각각 Y, Cb, 그리고 Cr 화소 성분을 위한 것으로 DWT 필터링의 동작 단계에 따라 해당 화소나 웨이블릿 계수들을 입력받는다. RAM 체인, 선-덧셈기, 부분곱 생성기(PP generator) 및 MUX, CSA 트리 그리고 최종 덧셈기까지 5단계의 파이프라인 단계를 거쳐서 하나의 화소에 대한 DWT 계수를 생성한다. 또한 하나의 DWT 계수를 만드는데 5번의 누적과정을 거치기 때문에 첫 DWT 계수가 출력되는데는 9 클록이 소요되고 다음 계수부터는 5 클록마다 하나씩 출력되어 전체적으로 DWT 커널은 5클록마다 4개의 계수가 출력된다.

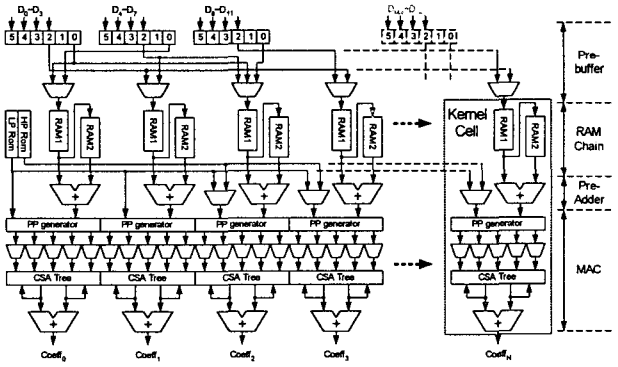

그림 2 DWT 커널과 커널 셀의 하드웨어 구조

Fig. 2 H/W architectures of DWT kernel and kernel cell

3.2. 양자화기/허프만 인코더의 H/W 구조

본 논문에서는 H/W의 복잡도 및 실시간 동작을 고려하여 양자화기는 선형 스칼라 양자화기를 사용하고 엔트로피 코딩은 허프만 코딩만을 이용한다. 허프만 코딩은 양자화 영역의 발생 빈도수로부터 허프만 코드를 발생시켜 두 과정을 하나의 통합된 구조로 결합하였고, 그 결과의 H/W 구조를 그림 3에 나타내었다. DWT 커널이 출력하는 계수들의 부대역 정보에 따라 양자화기가 두 부분으로 나누어져 있고, *FIFO*에서 정렬되어 허프만 코드값과 비트 정보 등을 출력한다. 양자화기 및 허프만 코더는 비교기와 ROM 테이블로 이루어져 있는데, 비교기에 의해 ROM의 주소가 결정되면 ROM으로

부터 허프만 코드("huff_code")와 유효 비트를 나타내는 비트 정보("bit_info")가 출력되고 이들과 함께 DWT 계수("wavelet_coeff")가 *FIFO*를 거쳐 출력된다. 5 클록마다 DWT 커널로부터 4개의 병렬 출력이 발생하므로 이를 순차적으로 처리하기 위해 FIFO가 시간적인 완충작용을 한다.

3.3. 역양자화기와 허프만 디코더의 H/W 구조

영상복원의 경우에서도 허프만 디코더는 역양자화기와 결합된 형태를 가지는데 그림 4에 보인 것과 같이 허프만 디코딩 결과로 16비트의 양자화 계수("Coefficient[15:0]")를 출력한다. 16비트 두 개의 입력 포트로부터 32비트 단위의 입력 데이터를 받아 코드경계 검출기(*Delimiter detector*)에 의해서 부대역 정보를 비롯한 영상 데이터를 추출하고 직렬 쉬프터(*HD shifter*)를 통해 직렬 데이터를 발생시켜 허프만 디코딩을 수행한다. 디코딩 시 예외 영역에 대한 정보가 검출되면 그 후의 4개의 직렬 데이터를 웨이블릿 계수로 변환하여 출력하는데, 양자화 시 빈도가 낮으면서 큰 값을 가지는 영역을 예외 영역으로 설정하고 허프만 코드와 함께 4비트의 계수 정보를 함께 포함시켰기 때문이다. 최저 주파수 대역의 데이터가 입력되면 허프만 디코딩을 거치지 않고 직렬 데이터들을 곧바로 웨이블릿 계수로 변환하여 출력하게 되는데, 이는 압축 과정에서 최저 주파수 대역의 정보를 정수부만 취하여 거의 무손실 압축을 하였기 때문이다.

3.4. 입력 인터페이스의 H/W 구조

그림 5에 보이는 입력 인터페이스는 NTSC 혹은 PAL 등의 표준 영상 형태를 입/출력하는 A/D 변환기(Bt829b)와 D/A 변환기(Bt866)와의 데이터 인터페이스를 위해서 사용한다. 또한 입력 인터페이스는 DWT 변환을 위해 입력 영상의 한 필드를 저장하기 위해 메모리를 이용할 경우 메모리와 입력 영상과의 버퍼링을 수행한다. 메모리를 사용하기 위하여 일반적인 방식의 메모리의 시동단계(power-up sequence)를 수행하는 회로(*200us counter*)를 내장하였고 A/D 변환기를 직접 연결하여 사용할 경우를 위하여 I^2C 제어기(*I^2C controller*)를 내장하고 있다. A/D 변환기를 사용할 경우 비디오 제어 신호 처리부(*Video Control Signal Process*)에서 비디오 제어신호를 분석하여 유효한 영상 데이터(Y, Cb, Cr data)만을 비디오 입력버퍼(*Video input buffer*)에 저장시켜 사용하고 비디오 제어신호 일부를 디코딩하여 내부적인 제어신호(Internal Video Control Signal)로 사용한다. 또한 외부

메모리를 사용할 경우, 일반적으로 메모리는 높은 클록 주파수를 가지는데 이를 위해 비디오 입력버퍼에 저장된 데이터를 고속 동작을 위한 메모리 입력버퍼(*Memory input buffer*)로 옮기고 "DatatoMem[15:0]" 포트를 이용하여 메모리 버퍼를 통해 메모리에 입력한다. 영상의 복원과정에서는 허프만 디코딩된 데이터("FromHuffDe")를 입력받아 메모리에 저장하는 동작을 수행하고 복원된 흑백과 색차 성분의 데이터("FromMem1[15:0]", "FromMeM2[15:0]")를 버퍼링하여 D/A 변환기로 출력("YtoDA[7:0]", "CtoDA[7:0]")한다.

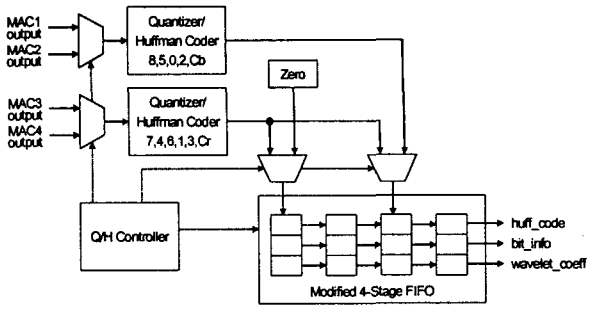

그림 3 양자화기/허프만 인코더의 하드웨어 구조

Fig. 3 H/W architecture of quantizer/Huffman encoder

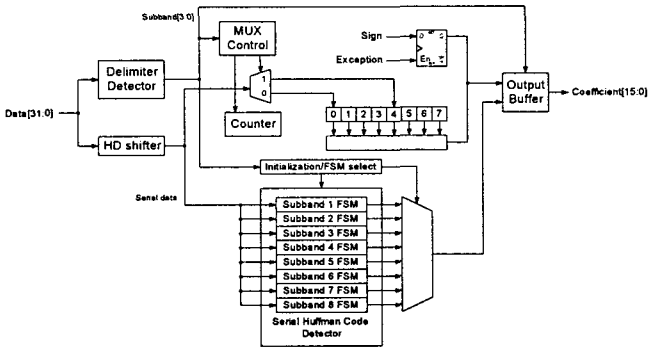

그림 4 허프만 디코더와 역양자화기의 하드웨어 구조

Fig. 4 H/W architecture of Huffman decoder and de-quantizer

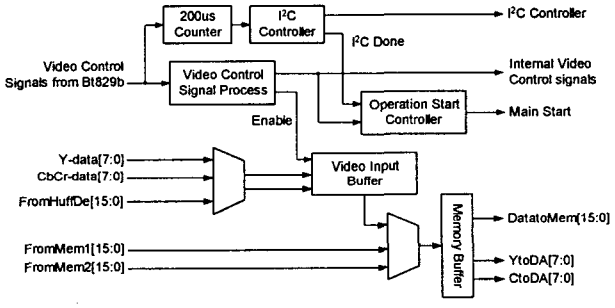

그림 5 입력 인터페이스의 하드웨어 구조

Fig. 5 H/W architecture of imput interface

3.5. 역변환 덧셈기/버퍼의 H/W 구조

영상의 복원과정에서 웨이블릿 역변환을 위한 필터링 수행 시 고주파 성분의 부대역과 저주파 성분의 부대역에 대한 필터링 결과를 더해야 복원된 웨이블릿 계수가 추출된다. 따라서 역변환을 위한 덧셈기와 그에 따른 시간적 완충이 요구되는데 이를 역변환 덧셈기/버퍼에서 수행하며, 그 구조를 그림 6에 나타냈다. 영상 압축과정에서는 이 덧셈과정이 필요 없으므로 그냥 지나친다. 각 변환 레벨("Level[3:0]")에 해당되는 제어신호("sel1", "sel2")에 따라서 "MACx_output"들의 조합을 이루어 더하거나 그냥 지나치게 하고 각 레지스터(*Reg*)에 저장하여 결과를 출력한다.

3.6. 출력 버퍼부의 H/W 구조

출력 버퍼부는 다양한 길이의 허프만 코드값("huff_code")을 32 비트의 일정 길이로 만드는 역할을 한다. 그림 7에서 보이는 것처럼 허프만 코드의 길이에 대한 정보는 비트정보 신호("bit_info")를 통해 얻을 수 있고 이를 이용해서 첫 번째 다중 쉬프터(*Multi-shifter1*)에 허프만 코드를 순서대로 저장하고 예외 영역에 해당되는 계수가 발생했을 경우는 허프만 코드 외에 웨이블릿 계수('wavelet_coeff")도 저장한다. 이와 동시에 비트 정보는 누적되면서 비교기(*Comparator*)에 의해 32비트 출력이 가능한지 판단한다. 32비트 이상의 허프만 코드의 입력이 확인되면 두 번째 다중 쉬프터(*Multi-shifter2*)에 보내지고 재정렬되어 32 비트 단위로 출력된다. 전체적인 동작은 부대역별로 이루어지고 32비트에서 부족한 데이터는 영값 확장(zero-padding)을 통해 32비트로 만든다. 영상의 시작과 끝, 부대역에 대한 정보, 그리고 필드 및 프레임 정보 등은 구획기(*Delimiter*)에 저장되어 압축결과와 함께 복원을 위한 정보가 부가된다.

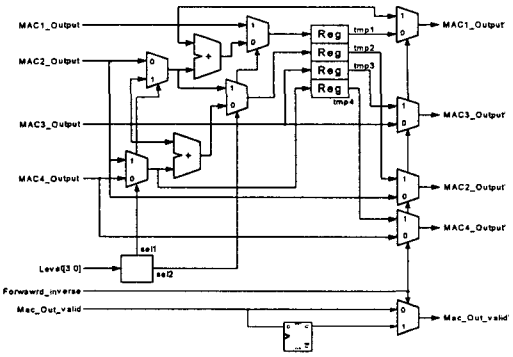

그림 6 역변환 덧셈기/버퍼의 하드웨어 구조

Fig. 6 H/W architecture of inverse adder/buffer

3.7. PLX 인터페이스부의 H/W구조

그림 8에 나타난 PLX 인터페이스부는 영상압축과정에서는 압축된 데이터를 외부로 출력하고 영상을 복원할 때에는 압축된 영상정보를 컴퓨터에 요구하여 데이터를 버퍼링하는 역할을한다. 영상의 압축과정에서는 출력 버퍼부에 의해 생성된 32 비트 단위의 허프만 코드("Data[31:0]")를 8단의 쉬프트 레지스터(shift register)를 통해 입력받아 병렬 레지스터에 채우고 양방향 버퍼/제어기(*Bi-dir Buffer&Ctrl*)를 통해 PLX9050으로 데이터("Lad[31:0]")를 출력한다. 영상의 복원 과정에서는 컴퓨터와 같은 저장매체로부터 입력된 데이터를 양방향 버퍼/제어기를 통해 쉬프트 레지스터에 입력하고 다시 허프만 디코딩을 위한 쉬프트 레지스터에 병렬 입력시킨 후 허프만 디코더의 호출에 따라서 32 비트 단위의 데이터("ToHuffmanDecoder[31:0]")를 출력시킨다. 또한 컴퓨터와은 호스트와의 제어통신을 통해서 압축/복원 혹은 영상/비디오 등의 전체적인 동작에 대한 정보를 전체 제어기에 제공하는데 이 경우에도 PLX9050을 통한 데이터 포트('Lad[31:0]")와 양방향 버퍼/제어기를 통해 데이터를 입력받아 프로그래머블 레지스터(*Programmable Register*)에 저장시킨 후 이를 디코딩하여 전체 동작을 제어하는데 이용한다.

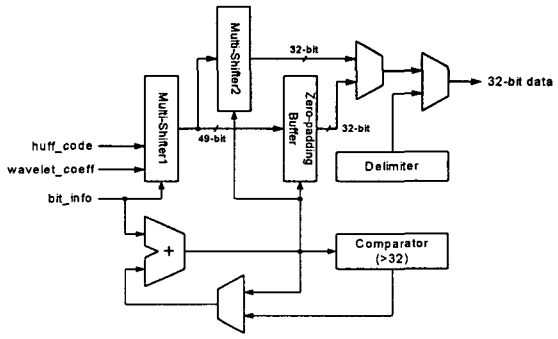

그림 7 출력 버퍼부의 하드웨어 구조

Fig. 7 H/W architecture of output buffer

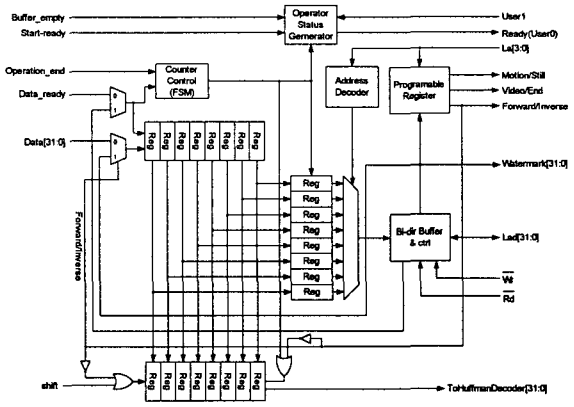

그림 8 PLX 인터페이스의 하드웨어 구조

Fig. 8 H/W architecture of PLX interface

4. 영상처리 H/W의 동작

본 장에서는 구현된 H/W의 압축과 복원을 위해 전체적으로 어떠한 과정을 거치는지를 나타내고 세부적인 동작 흐름에 대해서 설명한다.

4.1. 전체적인 H/W의 동작구성

제안된 H/W는 A/D 변환기의 필드 표시신호(그림 5에서 "Video Control Signals from BT829b"의 일부신호)에 동기하여 전체적인 동작이 제어되는데 영상압축 과정과 복원 과정에 대한 동작 순서를 그림 7과 8에 각각 나타냈다. 그림 7과 8에서 보듯이 필드단위로 영상을 처리하며, 그 시간은 약 15ms에 해당한다. 영상압축기의 데이터 처리량은 전적으로 A/D 변환기에서 출력되는 영상 데이터량에 의존하기 때문에 동작의 시작과 처리 시간을 A/D 변환기에 동기시킨다. 그림 7의 영상압축 과정은 크게 필드 단위의 영상을 메모리에 저장하는 과정(Even or Odd Field Store)과 영상을 압축하는 과정(Even or Odd Field Compression), 그리고 압축된 데이터를 출력(Even or Odd Field Q/H Hard disk Store)하는 과정으로 구성되는데, 이 세 가지 과정은 전체적으로 파이프라인 방식의 동작을 통해 실시간으로 영상을 처리한다. 또한 그림 8의 영상복원 과정은 압축된 데이터를 입력받아 허프만 디코딩하는 과정(Even or Odd Field HD and Store)과 복원

과정(Even or Odd Field De-compression), 그리고 복원된 영상을 출력하는 과정(Even or Odd Field D/A Converting)으로 구성되고 영상압축과정과 마찬가지로 전체적인 파이프라인 동작을 이룬다.

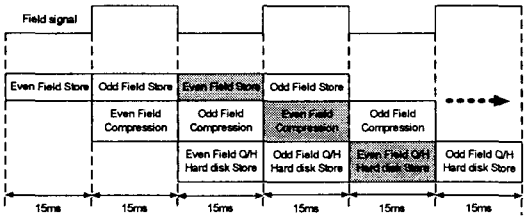

그림 9 영상 압축과정의 파이프라인 동작
Fig. 9 Pipeline operation of image compression

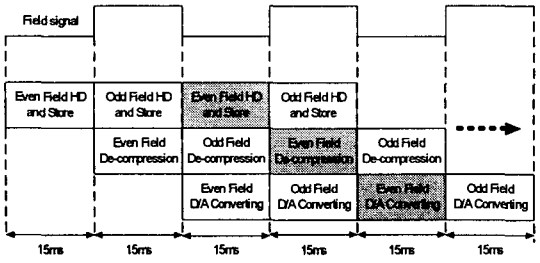

그림 10 영상 복원과정의 파이프라인 동작
Fig. 10 Pipeline operation of image reconstruction

4.2. 세부적인 영상압축과 복원 과정

그림 7과 8에서 나타난 전체적인 H/W의 동작에서 영상의 압축과 복원과정은 세부적으로 표 2에 나타난 것과 같이 크게 세가지 모드(YCbCr, Y-only, CbCr)를 가지고 총 16단계의 과정을 거친다. 각각의 모드에 따라서 그림 2의 프리-버퍼에 각기 다른 영상 데이터가 입력되고 수직(Col) 혹은 수평방향(Row) 필터링 과정을 16단계, 즉 4 레벨(Level) 거쳐서 영상압축과 복원 과정을 수행한다. 내부적으로 전체 제어기는 "State" 신호를 이용해서 개별적인 과정에 대한 정보를 전체 H/W에 알려주고 전체적으로 제어한다.

영상압축과 복원에 대한 세부동작을 그림 11에 도식하였다. 3.4절에서 설명한 것과 같이 초기에 외부 메모리의 시동동작(memory power sequence)과 A/D 변환기의 초기 설정을 위한 동작(I^2C) 후 메모리에 필드 영상을 입력받을 수 있는 상태임(Start ready='1')을 외부에 알려준다. 다음에 S/W의 동작 시작 명령을 관찰한 후 시작신호가 인지(start='1')되면 영상압축 혹은 복원을 시작하는데, 먼저 필드단위로 메모리에 영상을 저장하고 알맞은 시점에서 영상압축 혹은 복원을 필드 단위로 수행된다. 영상복원 과정이 선택되었다면 압축된 데이터를 입력받아서 허프만 디코딩을 거치고 메모리에 저장한다. 영상압축과 복원과정은 16단계로 구성되어 있는데, 각 단계마다 처리되는 계수의 전체 개수를 조사하기 때문에 데이터 소실을 통한 부대역간의 에러 전파가 발생하지 않는다. 영상압축 및 복원과정을 끝낼 때는 최종 레벨(Final level) 수행 후 홀수 필드(Odd field)인지를 검사하여 동작을 멈춘다.

표 2 세부적인 영상압축과 복원과정 구성
Table 2 Image compression and re-construction procedure in detail

		Compression			De-Compression		
#	State	Mode	Filter	Level	Mode	Filter	Level
0	0000	Ready / Initial			Ready / Initial		
1	0001	Y,Cb,Cr	Row	1	Y-only	Col	4
2	0010	Y-only	Col			Row	
3	0011		Row	2		Col	3
4	0100		Col			Row	
5	0101		Row	3		Col	2
6	0110		Col			Row	
7	0111		Row	4		Col	1
8	1000		Col		Cb,Cr	Col	4
9	1001	Cb,Cr	Col	1		Row	3
10	1010		Row	2		Col	2
11	1011		Col			Row	
12	1100		Row	3		Col	1
13	1101		Col			Row	
14	1110		Row	4		Col	1
15	1111		Col		Y,Cb,Cr	Row	

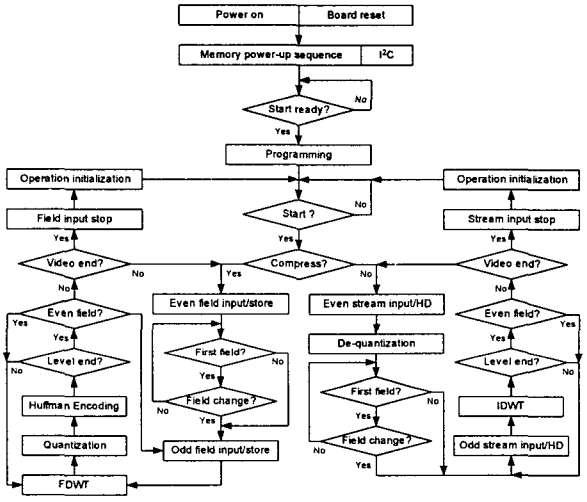

그림 11 영상압축과 복원 과정의 세부적인 흐름도
Fig. 11 Image compression and reconstruction flow in detail

5. 제안된 H/W의 구현결과

본 논문은 DWT을 이용하여 영상을 압축 및 복원하는 H/W를 구현하고 FPGA상에서 구현하는 것을 목적으로 하였다. 본 장에서는 H/W로 구현한 뒤의 FPGA 상에서 구현한 결과와 H/W 자원 사용률 등을 보이고 압축 후 복원된 영상을 간단히 나타낸다.

2, 3장을 통해 소개한 H/W구현을 위해 Altera의 Quartus II 환경을 사용하고 VHDL을 이용하여 설계하였다. 타겟 플랫폼으로 이용된 FPGA는 Altera의 PEX20KC EP20K1000CB652-7칩이다. 그림 12에 Quartus II 환경의 그래픽 에디터(GDF)로 설계된 전체 블록도를 나타냈다. 세부

적인 회로는 RTL(Register Transfer Level) 수준으로 설계하고 전체적으로는 구조적 수준으로 취합하였다. 내부적 동작 메모리로 FPGA 내부의 일부 ESB(Embedded System Block)을 사용하였는데 ASIC(Application Specific Integrated Circuit)화 할 경우 ESB를 해당 ASIC 라이브러리에 맞는 메모리 모델로 교체하면 되므로 FPGA을 기본적인 설계 플랫폼으로 삼았다 할지라도 구현된 H/W는 범용성을 가질 수 있다. 그림 1과의 비교를 용이하게 하고자 블록들을 동일하게 표기하였고 유사하게 배치하고자 하였다. 단, 양자화와 허프만 인코더가 분리되어 그림 2와 다른 것을 볼 수 있는데 이것은 양자화기의 양자화 인덱스를 관찰하기 위해 분리한 것으로 동작 혹은 구조가 변화한 것은 아니다.

구현된 H/W는 33MHz(30ns의 주기)의 기본 주파수에서 동작하여 초당 약 67필드(33프레임)의 영상을 압축할 수 있기 때문에 실시간 동작이 가능하다. 이 때 메모리는 약 100MHz 클록(99MHz, 기본 동작주파수의 3배 클록 속도)에 의해서 동작해야 하는데 두 클록간의 동기를 위해서 APEX FPGA 칩이 내부적으로 제공하는 클록 가속기(clock boosting) 기능을 이용하였다. 그리고 입력 인터페이스는 A/D 변환기의 동작 주파수인 28.63636MHz에 맞추어서 동작한다.

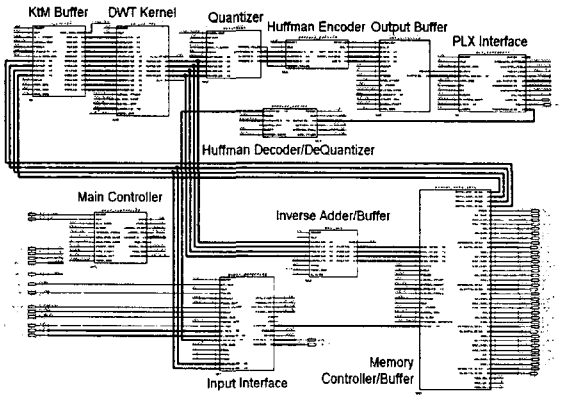

그림 12 Quartus II를 이용한 그래픽 설계 결과
Fig. 12 GDF design result using Quartus II

표 3에서는 그림 1과 12의 각 기능 블록들이 APEX20KCEP20K600CB652-7에 사상되었을 때 사용된 H/W 자원 사용율을 나타냈다. 전체적으로 타겟 FPGA의 53%(12943개)의 LAB(Logic Array Block)를 사용하고 9%(28352개)의 ESB를 사용한다. 메모리 제어기의 경우 메모리 호출 주소 중 많은 부분을 ROM에 저장하여 사용하므로 나수의 ESB를 사용한다. 또한 커널 내부에서 커널을 제어하는 커널 제어부도 제어에 필요한 동작을 ROM에 저장하기 때문에 ESB를 사용하게 된다. 전체 제어부(Main Controller)의 자원 사용률이 상대적으로 낮은데, 이는 각각의 기능 블록들이 내부에 따로 제어부를 가지고 있어 각 블록들이 자체 제어부에 의해 제어가 되기 때문이다. 즉, 전체 제어부는 모든 블록들의 내부 제어부들의 전체적인 동작 순서만을 결정한다. 출력 버퍼 블록은 비교적 많은 H/W 자원을 사용하는데, 이는 허프만 코더에서 불규칙한 크기의 데이터를 축적하고 있다가 32 비트로 만들어 출력시키는 부분에서 많은 병렬 레지스터와 쉬프트 레지스터를 사용하기 때문이다.

그림 13은 약 40:1의 압축률로 압축 과정을 거친 컬러 영상에 대해 다시 복원한 후의 결과를 보이고 있는데 (a)는 원래의 필드 영상을 나타내고 (b)는 복원된 필드 영상을 나타낸다.

표 3 구현된 하드웨어의 FPGA 자원 사용률
Table 3 FPGA resource usage of the implemented H/W

Block Name	Logic Array Block(24320:100%)	Embedded System Block(311296:100%)
DWT kernel	3129 (13%)	9472 (3%)
Quantizer	975 (4%)	0 (0%)
Huffman encoder	283 (1%)	0 (0%)
Output buffer	2405 (10%)	0 (0%)
PLX interface	1433 (6%)	0 (0%)
KtM buffer	1242 (5%)	0 (0%)
Main controller	168 (<1%)	0 (0%)
Huffman decoder/ Dequantizer	1154 (5%)	0 (0%)
Input interface	391 (2%)	0 (0%)
Memory controller /Buffer	1501 (6%)	18880 (6%)
Inverse adder/buffer	262 (1%)	0 (0%)
Total	12943 (53%)	28352 (9%)

(a) (b)

그림 13 압축 후 복원된 Lena 영상 결과 (a) 원래 영상 (b) 복원된 영상
Fig. 13 Reconstructed Lena image after compression (a) original image (b) reconstructed image

5. 결 론

본 논문에서는 입력되는 영상 혹은 압축 데이터를 실시간으로 압축하거나 복원할 수 있는 H/W의 구조를 제안하고 FPGA를 타겟으로 하여 구현하였다. Motion JPEG2000을 고려하여 공간영역의 데이터를 주파수 영역으로 변환하는 도구로서 DWT를 채택하여 컨벌루션 방식의 확장이 용이한 필터링 구조를 제안하였다. 시각적 특성을 토대로 통계적으로 구성한 양자화와 허프만 코딩과정을 통합된 형태의 H/W로 모두 내장시켰고 영상의 표준 입출력을 처리할 수 있는 입출력 장치들도 내장하였다. 구현된 H/W는 영상압축과 복원과정을 모두 수행가능한데 전체 동작은 A/D 변환기에 동기하고 필드단위의 동작을 파이프라인 방식으로 수행하여 실시간으로 동작하기 위한 충분한 성능을 보였다.

구현된 H/W는 Altera의 APEX20KC EP20K600CB652-7

FPGA 칩에서 53%(12943개)의 LAB와 9%(28352개)의 ESB을 사용하면서 문제없이 사상되었다. 그리고 약 70MHz의 클록주파수에서 안정적으로 동작할 수 있어 기준 클록으로 가정하였던 33MHz의 속도를 충분히 만족시켰다. 따라서 초당 60필드(30 프레임) 이상으로 영상을 처리를 할 수 있어 실시간 처리가 가능함을 확인하였다.

JPEG2000을 비롯하여 웨이블릿 변환 기반 제품의 저변이 확대되고 있는 시점에서 구현된 하드웨어는 독립적으로 하나의 영상처리 제품이 될 수 있고 SOC(system-on-a-chip)를 구성하는 IP(intellectual property)로도 사용될 수 있을 것으로 사료된다. 또한 향후 본 연구는 단일 형태의 Motion JPEG2000 칩셋으로 더욱 접근하기 위해 EBCOT(Embedded Block Coding with Optimized Truncation), ROI(region of interest), 그리고 적응적인 선형 양자화 알고리즘들을 H/W에 적합하게 개발하여 구현된 H/W의 기능 블록들로 추가할 계획이다.

감사의 글

이 논문은 2003년도 광운대학교 교내 학술연구비 지원에 의해 연구되었음.

참 고 문 헌

[1] G. Knowles, "VLSI Architectures for the Discrete Wavelet Transform", *IEEE Electronic Letters*, Vol. 26, No. 15, pp. 1184-1185, July 1990.

[2] A.S. Lewis and G. Knowles, "VLSI Architecture for 2-D Daubechies Wavelet Transform without Multipliers", *IEEE Electronic Letters*, Vol. 27, No. 2, pp. 171-173, Jan. 1991.

[3] J. Fridman and E. S. Manolakos, "Distributed Memory and Control VLSI Architectures for 1-D Discrete Wavelet Transform", *IEEE Workshop on Signal Processing Systems*, pp. 388-397, 1994.

[4] T.K. Truong, K.C. Hung, Y.J. Haung, and Y.H. Tseng, "A New Architecture for the 2-D Discrete Wavelet Transform", *IEEE Int'l Conf. of Communications Computers and Signal Processing*, pp. 481-484, 1997.

[5] C. Yu and S.J. Chen, "Design of an Efficient VLSI Architecture for 2-D Discrete Wavelet Transform", *IEEE Trans. on Consumer Electronics*, Vol. 45, No. 1, pp. 135-140, Feb. 1999.

[6] M.H. Sheu, M.D. Shieh and S.W. Liu, "A VLSI Architecture Design with Lower Hardware Cost and Less Memory for Separable 2-D Discrete Wavelet Transform", *IEEE ISCAS'98*, Vol. 5, pp. 457-460, 1998.

[7] M. Vishiwanath, R. Michael and M.J. Irwin, "BSLI Architecture for the Discrete Wavelet Transform", *IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing*, Vol. 42, No. 5, pp. 305-316, May 1995.

[8] J. Chen and M.A. Bayoumi, "A Scalable Systolic Array Architecture for 2-D Discrete Wavelet Transforms", *IEEE Proc. of Midwest Symp. on Circuits and Systems*, Vol. 2, pp. 303-312, 1996.

[9] W. Ravasi, M. Mattavelli, D.J. Mlynek, A. Butter, and A Soudagar, "Wavelet Image Compression for Mobile/Portable Applications", *IEEE Trans. on Consumer Electronics*, Vol. 45, No. 3, pp. 794-803, Aug. 1999.

[10] S. Masud and J.V. McCanny, "Wavelet Packet Transform for System-on-Chip Application", *IEEE Proc. on ICASSP*, Vol. 6, pp. 3287-3290, 2000.

[11] A.M. Reza and R.D. Turney, "FPGA Implementation of 2D Wavelet Transform", *IEEE Conf. of Signals, Systems and Computers*, pp. 584-588, 1999.

[12] W. Jiang and A. Ortega, "Lifting factorization-based discrete wavelet transform architecture design," IEEE Trans. Circuits Syst. Video Technol., vol. 11, pp. 651-657, May 2001.

[13] C. Diou, L. Torres, and M. Robert, "A wavelet core for video processing," presented at the IEEE Int. Conf. Image Process., Sept. 2000.

[14] G. Lafruit, L. Nachtergaele, J. Bormans, M. Engels, and I. Bolsens, "Optimal memory organization for scalable texture codecs in MPEG-4, IEEE Trans. Circuits Syst. Video Technol., vol. 9, pp. 218-243, Mar. 1999.

[15] K. Andra, C. Chakrabarti, and T. Acharya, "A VLSI architecture for lifting-based forward and inverse wavelet transform," IEEE Trans., on Signal Processing, vol. 50, no. 4, Apr 2002

[16] M. Ravasi, L. Tenze, and M. Mattavelli, "A scalable and programmable architecture for 2-D DWT decoding," IEEE Trans. Circuits Syst. Video Technol., vol. 12, no. 8, Aug. 2002

[17] A. Said and W.A. Pearlman, "A New, Fast and Efficient Image codec Based on Set Partitioning in Hierarchical Trees", *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 6. No. 3, pp. 243-250, June 1996.

[18] S.F. Hsiao, Y.C. Tai and K.H. Chang, "VLSI Design of an Efficient Embedded Zerotree Wavelet Coder with Function of Digital Watermarking", *IEEE Trans. on Consumer Electronics*, Vol. 46, No. 3, pp. 628-636, August 2000.

[19] B. Parhami, "Computer Arithmetic : Algorithm and Hardware Desig", Oxford University Press, 2000.

저 자 소 개

서 영 호 (徐 英 鎬)

1999년 2월 : 광운대학교 전자재료공학과 졸업(공학사). 2001년 2월 : 광운대학교 대학원졸업(공학석사). 2000년 3월~2001년 12월 : 인티스닷컴(주) 연구원. 2001년 3월~현재 : 광운대학교 전자재료공학과 박사과정. 2003년 6월~현재 : 한국전기연구원 연구원. <관심분야> Image Processing/Compression, 워터마킹, 암호학, FPGA/ASIC 설계

Tel : 02-940-5167, Fax : 02-919-3940

E-mail : design@kw.ac.kr

김 동 욱 (金 東 郁)

1983년 2월 : 한양대학교 전자공학과 졸업(공학사). 1985년 2월 : 한양대학교 대학원 졸업(공학석사). 1991년 9월 : Georgia 공과대학. 전기공학과 졸업(공학박사). 1992년 3월~현재 : 광운대학교 전자재료공학과 정교수. 광운대학교 신기술 연구소 연구원. 1997년 12월~현재 : 광운대학교 IDEC 운영위원. 2000년 3월~현재 : 인티스닷컴(주) 연구원. <관심분야> 디지털 VLSI Testability, VLSI CAD, DSP 설계, Wireless Communication

Tel : 02-940-5167, Fax : 02-919-3940

E-mail : dwkim@daisy.gwu.ac.kr