논문 2009-46SD-7-5

JPEG2000 이산웨이블릿변환의 컨볼루션기반 non-cascaded 아키텍처를 위한 pipelined parallel 최적화 설계

(A Pipelined Parallel Optimized Design for Convolution-based Non-Cascaded Architecture of JPEG2000 DWT)

이 승 권*, 공 진 흥**

(Seung-Kwon Lee and Jin-Hyeung Kong)

요 약

본 연구에서는 실시간 이산웨이블릿변환을 위한 컨볼루션기반 non-cascaded 구조를 구현하고자 병렬곱셈기-중간버퍼-병렬누적기의 고성능 병렬파이프라인 연산회로를 설계하였다. 이산웨이블릿변환의 컨볼루션 곱셈연산은 필터계수의 대칭성과 업/다운 샘플링이 고려된 최적화를 통해서 1/4정도로 감소시킬 수 있으며, 화상데이터와 다수 필터계수들 간의 곱셈과정을 LUT기반의 병렬계수 DA 곱셈기 구조로 구현하면 3~5배 고속연산처리가 가능하게 된다. 또한 컨볼루션의 곱셈결과를 중간버퍼에 저장하여 누적가산 과정에서 재사용하면 전체 곱셈연산량을 1/2로 감소시켜 연산전력을 절약시킬 수 있다. 중간버퍼는 화상데이터와 필터계수들의 곱셈결과값들을 컨볼루션의 누적가산 과정을 위해 정렬시켜 저장하게 되는데, 이때 병렬누적가산기의 고속 순차검색을 위해 정렬된 병렬저장이 이루어지도록 버퍼관리 구조를 설계한다. 컨볼루션의 병렬곱셈기와 병렬누적가산기는 중간버퍼를 이용한 파이프라인을 구성하게 되는데, 파이프라인 연산처리 효율을 높이기 위해 병렬곱셈기의 연산처리 성능에 맞추어 누적가산기 및 중간버퍼의 병렬화 구조가 결정된다. 설계된 고성능 이산웨이블릿변환기의 성능을 검증하기 위해서 0.18um 라이브러리를 이용한 후반부 설계를 하였으며, 90MHz에서 SVGA(800x600)영상을 30fps로 실시간 처리함을 확인하였다.

Abstract

In this paper, a high performance pipelined computing design of parallel multiplier-temporal buffer-parallel accumulator is present for the convolution-based non-cascaded architecture aiming at the real time Discrete Wavelet Transform(DWT) processing. The convolved multiplication of DWT would be reduced upto 1/4 by utilizing the filter coefficients symmetry and the up/down sampling; and it could be dealt with 3-5 times faster computation by LUT-based DA multiplication of multiple filter coefficients parallelized for product terms with an image data. Further, the reutilization of computed product terms could be achieved by storing in the temporal buffer, which yields the saving of computation as well as dynamic power by 50%. The convolved product terms of image data and filter coefficients are realigned and stored in the temporal buffer for the accumulated addition. Then, the buffer management of parallel aligned storage is carried out for the high speed sequential retrieval of parallel accumulations. The convolved computation is pipelined with parallel multiplier-temporal buffer-parallel accumulation in which the parallelization of temporal buffer and accumulator is optimized, with respect to the performance of parallel DA multiplier, to improve the pipelining performance. The proposed architecture is back-end designed with 0.18um library, which verifies the 30fps throughput of SVGA(800x600) images at 90MHz.

Keywords : JPEG 2000, DWT, pipelined parallel architecture, non-cascaded, convolution-based

* 정회원, (주) 동운아나텍 연구개발본부
(R&D Division, Dongwoon Anatech Co.,Ltd.)
** 평생회원, 광운대학교 컴퓨터공학과
(Department of Computer Engineering, Kwangwoon University)
※ "본 논문은 2009년도 광운대학교 교내학술연구비 지원과 2007년도 광운대학교 연구년에 의해 연구되었으며, 2009년도 「서울시 산학연 협력사업」과 ETRI 「IT SoC 핵심설계인력양성사업」 결과이며, 시스템집적반도체 기반 기술개발사업과 반도체설계교육센터(IDEC)의 지원으로 이루어졌습니다."
접수일자: 2008년12월16일, 수정완료일: 2009년6월30일

Ⅰ. 서 론

최근 VOD, HDTV, DVD 등 멀티미디어 응용 분야에서 영상데이터를 압축하는 JPEG, MPEG 등의 표준에 대한 관련 연구들이 활발하게 진행되고 있다. JPEG 또는 MPEG은 블록기반의 이산여현변환(Discrete Cosine Transform)을 사용하는 기술인데, 압축률이 높아짐에 따라 블록의 경계부분에 열화 현상을 발생시키는 구획화 현상(Blocking effect)이 나타나는 한계를 갖는다. 반면에 하나의 프레임 단위로 영상을 처리하는 이산웨이블릿 변환(Discrete Wavelet Transform)은 구획화 현상을 발생시키지 않아서 JPEG2000의 영상압축 표준기술로 사용되고 있다. 그러나 2차원 프레임 단위의 이산웨이블릿변환은 고대역/저대역 필터 연산을 반복하여 수행하기 때문에 연산량이 많아서 실시간 고속처리와 저전력 소모를 위한 VLSI 하드웨어 구현이 필요하다.

이산웨이블릿 변환기의 하드웨어 구현에 관한 기존 연구는 리프팅 기반의 구조[1]와 컨볼루션 기반의 구조[2-4]에 관한 연구로 구분된다. 리프팅 기반의 구조는 컨볼루션 기반의 구조에 비해 곱셈연산량을 줄일 수 있다. 그러나 하드웨어 구조가 복잡하고 임계 경로(critical path)가 길기 때문에 초기 지연시간이 길어져 계산속도가 늦어지는 문제를 갖는다. 컨볼루션 기반의 구조는 리프팅기반 구조에 비해 곱셈연산량은 많지만 지연시간이 짧아 계산속도가 빠르다. 컨볼루션 기반의 cascaded 구조[2]는 레벨이나 라인단위를 다중 프로세싱 유닛으로 동시에 연산을 처리하여 계산속도는 빠르다. 그러나 각각의 프로세싱 유닛은 필터계수 길이만큼의 곱셈기를 가지며 하드웨어 복잡도가 증가하는 단점이 있다. 컨볼루션 기반의 non-cascaded 구조[3~4]는 레벨이나 라인단위의 연산을 하나의 프로세싱 유닛으로 동작하며 제어와 구조가 간단한 장점을 갖지만 cascaded 구조에 비해 계산속도는 느리다. 컨볼루션 기반의 non-cascaded 구조는 다시 시스톨릭 구조[3]와 병렬 구조[4]로 나뉜다. 시스톨릭 구조는 병렬 구조에 비해 계산속도가 빠르지만 지연시간이 길고 파이프라인 동작이 어려운 반면, 병렬 구조는 시스톨릭 구조에 비해 계산속도는 느리지만 지연시간이 짧고 하드웨어 복잡도가 상대적으로 간단하며 파이프라인 동작이 용이하다.

이산웨이블릿 변환을 위한 non-cascaded 병렬구조는 크기가 작고 구조가 간단하며 효율적인 파이프라인 동작이 가능하여 고속연산에 적합한 구조를 가지지만, 컨볼루션 기반은 곱셈연산량이 많은 문제점을 가지고 있다. 본 연구에서는 2차원 정방향 및 역방향 이산웨이블릿변환기를 컨볼루션기반/non-cascaded/병렬구조로 설계하며, 크기가 작고 구조가 간단하며 효율적인 파이프라인의 고속연산처리가 가능하도록 구현하였다. 컨볼루션 기반의 이산웨이블릿 변환이 갖는 단점인 많은 곱셈연산량을 해결하고자, Daubechies(9/7) 필터의 필터계수 대칭성을 이용하여 동일한 값을 지니는 필터계수와의 곱셈연산을 버퍼에 저장 및 재사용하여 곱셈연산량을 줄였으며, 다운(업) 샘플링과정에서 불필요한 샘플과 필터계수와의 곱셈연산을 생략하였다. 또한 컨볼루션 연산에서 성능에 많은 영향을 미치는 곱셈연산을 고속처리하기 위해 입력샘플과 필터계수와의 곱셈연산 알고리즘을 병렬화하였고, LUT기반의 고속곱셈 연산회로를 설계하였다. 그리고 이산웨이블릿 변환기의 컨볼루션 연산을 곱셈&저장과 검색&덧셈의 2단 파이프라인 동작으로 구분하고, 곱셈&저장과 검색&덧셈의 처리성능이 같도록 설계하여 컨볼루션 파이프라인의 효율을 높이고자 하였다. 이를 위해서 병렬설계된 컨볼루션 고속 곱셈부와 균형적인 컨볼루션 덧셈부의 병렬화 계수를 탐색하였고, 검색시간을 줄이기 위해서 덧셈연산 과정에 필요한 데이터를 순차적으로 검색할 수 있는 고속 정렬(alignment) 버퍼를 설계하였다. 또한 이산웨이블릿 변환기의 곱셈부 및 덧셈부의 연산정확도에 따른 PSNR 값의 변화를 분석하여 연산 양자화 오류를 최소화할 수 있는 최적의 데이터 패스폭을 설계하였다.

제 Ⅱ장에서는 DWT/IDWT 컨볼루션 기반 non-cascaded 병렬 구조의 연산성능 및 정확도를 위한 최적화 설계방법을 기술하고, 제 Ⅲ장에서는 DWT/ IDWT를 곱셈&저장과 검색&덧셈으로 2단 파이프라인 구조로 설계하고 각 파이프라인 단계의 처리성능이 같도록 병렬화하여 컨볼루션 파이프라인의 효율을 높였다. 제 Ⅳ장에서는 DWT/IDWT의 비트폭에 따른 영상 압축/복원 성능을 검증하고, Altera/Xilinx FPGA와 ASIC (0.18um)으로 설계하여 DWT/IDWT의 구현 성능을 확인하였다. 마지막 Ⅴ장에서 본 연구의 결론을 기술한다.

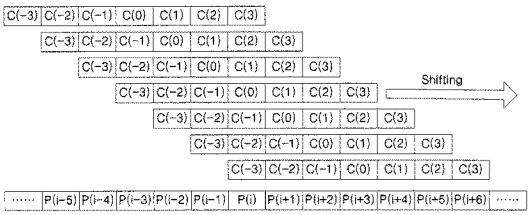
Ⅱ. DWT/IDWT 컨볼루션 연산처리

1. 연산 성능

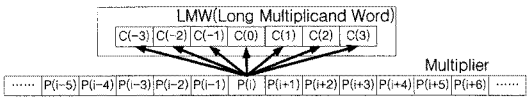
가. 곱셈연산 최적화

이산웨이블릿 변환기의 컨볼루션 연산은 상수형 필터계수와 변수형 샘플데이터와의 곱셈과정을 포함한다. 승수나 피승수중 하나가 고정된 상수형계수곱셈기(Constant Coefficient Multiplier, KCM)는 분산산술(Distributed Arithmetic, DA) 구조로써 효율적으로 구현된다. DA 방식의 곱셈기 구조는 입력 영상데이터와 고정된 필터계수의 곱셈값을 ROM 테이블에 저장시키고, 영상 데이터 값을 입력 주소로 하여 LUT(Look Up Table) ROM을 검색하여, 곱셈값을 고속 연산하는 것이 가능하다.

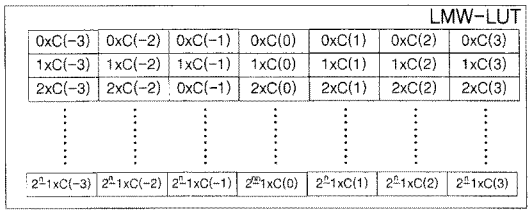
이산웨이블릿 변환에서 그림 1(a)의 컨볼루션 연산은 필터계수와 입력데이터간의 곱셈 동작을 반복하면서 하나의 데이터가 모든 필터계수들과 곱셈연산을 하게 된다. 이때 입력샘플과 곱셈연산을 수행하는 다수의 필터계수들을 병렬화시킨 하나의 피승수 (Long Multiplicand Word, LMW) {C(-3)|C(-2)|C(-1)|C(0)|C(1)|C(2)|C(3)}를 그림 1(b)와 같이 구성하면 LMW와 입력 데이터와의 병렬 곱셈결과를 LUT의 한주소에 저장하는 것도 가능하다. 따라서 LMW와 입력데이터 P(i)와의 곱셈결과 {P(i)C(-3)|P(i)C(-2)|…|P(i)C(3)}를 그림 1(c)와 같이 저장하면 컨볼루션 연산의 병렬곱셈 LUT를 설계할 수 있다. 또한 컨볼루션 연산의 Daubechies (9,7) 필터계수의 대칭성을 이용하면 LUT의 병렬곱셈값을 {P(i)C(0)|P(i)C(1)|P(i)C(2)|P(i)C3)}로 구성하여 LUT의 데이터 패스폭을 그림 1(d)와 같이 최적화시킬 수 있다. 정방향 이산웨이블릿 과정은 2차원 영상 이미지에 대한 고대역 필터 연산과 저대역 필터 연산을 처리하고 다운 샘플링을 수행한다. 다운 샘플링 과정에서 고대역 필터연산의 경우 홀수번째 데이터만 덧셈연산하고, 저대역 필터연산의 경우 짝수번째 데이터와 덧셈연산하는 것을 알 수 있다. 따라서 그림 2와 같이 컨볼루션 덧셈연산에 사용되지 않는 곱셈 결과값들(어두운색 영역)을 제거하면 곱셈 결과값을 저장하기 위한 메모리 크기를 1/2로 최적화시킬 수 있다. 이와 같은 방법으로 역방향 웨이블릿 변환의 과정의 업 샘플링 과정도 불필요한 곱셈결과를 제거하면 곱셈결과 값의 메모리 크기를 반으로 줄일 수 있다.



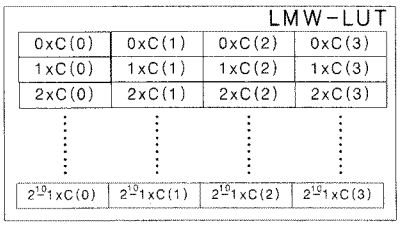
(a) 컨볼루션 곱셈과정



(b) 필터계수의 병렬화



(c) 병렬곱셈 LUT(S=2n×7)



(d) 병렬곱셈 LUT 최적화(S=2n×4)

그림 1. LMW LUT의 설계 및 최적화
Fig. 1. LMW LUT design and optimization.

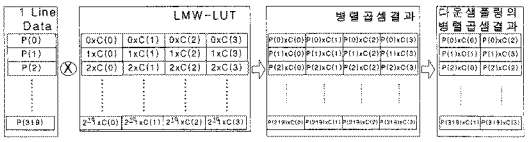


그림 2. 다운샘플링에 따른 병렬 곱셈량의 감소
Fig. 2. Reduced parallel multiplication of downsampling.

나. 파이프라인 설계

(1) 곱셈&저장과 검색&덧셈의 2단 파이프라인

컨볼루션 필터연산은 입력 영상데이터마다 모든 필터계수들과 곱셈결과를 계산하고, 생성된 곱셈결과들에 대한 누적덧셈연산으로 이루어진다. 따라서 컨볼루션 필터연산은 그림 3과 같이 곱셈연산과 덧셈연산간의 2단 파이프라인 동작을 라인단위로 처리해서 연산처리 성능을 향상시킬 수 있다. 실제로 라인단위의 샘플데이

곱셈&저장(i)	곱셈&저장(i+1)	곱셈&저장(i+2)
검색&덧셈(i-1)	검색&덧셈(i)	검색&덧셈(i+1)

그림 3. 라인단위 2단 파이프라인 컨볼루션 연산
Fig. 3. 2-stage pipelined convolution by line unit.

터를 필터계수들과 병렬곱셈하여 곱셈결과들을 생성, 버퍼에 저장한다. 그리고 버퍼에 저장된 라인단위의 곱셈결과들을 검색해서 덧셈연산으로 컨볼루션 연산결과를 얻게 되는데, 이와 동시에 새로운 라인단위의 샘플 데이터를 필터계수들과 병렬곱셈하여 버퍼에 인터리빙 저장한다.

(2) 파이프라인 실행효율

곱셈&저장 단계에서는 픽셀데이터와 LMW 필터계수와의 한번의 병렬곱셈으로 픽셀데이터의 컨볼루션에 필요한 모든 곱셈결과들을 생성한다. 덧셈&저장단계에서 고대역 및 저대역 필터의 연산횟수는 다운샘플링에 의해 픽셀데이터 당 각각 7/9회에서 3.5/4.5회로 절반으로 감소하였으나, LMW 병렬곱셈연산의 1회에 비해 많은 연산이다. 따라서 파이프라인의 데이터 처리성능을 높이기 위해 덧셈부의 병렬화가 요구된다. 그림 4는 고대역/저대역 필터 연산에서 곱셈연산과 덧셈연산의 파이프라인 동작을 덧셈연산 병렬화 계수 2/4/8/12/16에 대하여 보여주고 있다. 병렬화 계수 2는 고대역/저대역 필터 연산에서의 덧셈연산을 각각 병렬 연산함을 의미하며, 병렬화 계수 4는 고대역 필터연산을 위해 2개의 병렬 덧셈과 저대역 필터연산을 위해 2개의 병렬 덧셈을 연산하는 것을 의미한다. CIF 1 라인 = 352 샘플데이터에서의 파이프라인 주기를 살펴보면 덧셈연산의 병렬화 계수가 2에서 8까지 증가함에 따라 파이프라인 주기가 짧아져 파이프라인의 데이터 처리성능이 향상됨을 알 수 있다. 그러나 덧셈연산 병렬화 계수가 12이상이 되면, 곱셈연산으로 병목현상이 이동되어 파이프라인 주기는 더 이상 짧아지지 않는다. 효율적인 파이프라인 동작을 살펴보고자 표 1과 같이 덧셈연산 병렬화 계수에 따른 전체 파이프라인 주기 사이클에서 파이프라인이 실제로 동작하는 사이클이 차지하는 비율을 비교하였다. 실행효율이 1에 가까울수록 각 파이프라인단계가 "IDLE"상태 없이 효율적인 파이프라인 동작을 하게 된다. 비교한 결과 병렬화 개수 8일 때 실행효율이 0.94로 효율적인 파이프라인을 하고 있음을 알 수 있다.

2. 데이터패스 폭

JPEG 2000의 부호화/복호화 과정에서 압축 및 복원된 이미지는 양자화오류와 정방향/역방향 이산웨이블릿 변환 과정의 연산오류에 의해서 화질이 저하될 수 있다. 따라서 정방향/역방향 이산웨이블릿 변환의 연산오류를 최소화하면서 연산기의 크기를 작게 설계하기 위해서 정방향/역방향 이산웨이블릿 변환기의 데이터패스 폭에 따른 압축 및 복원된 영상의 PSNR을 압축률(35:1)에서 비교하여 그림 5와 같이 보였다. Lena, Babara, Boats 등 이미지에 대한 정수 C 시뮬레이션 결과에 따르면, 정방향/역방향 이산웨이블릿 변환의 정수 데이터 비트 범위는 10 비트까지 30dB 이상의 고화질 영상을 유지함을 알 수 있다. 따라서 고화질 영상과 최소의 데이터패스를 갖는 정방향/역방향 이산웨이블릿 변환기의 데이터패스폭은 10비트로 설계되어야 한다.

III. DWT/IDWT 아키텍처 설계

1. 정방향/역방향 이산웨이블릿 변환기 통합 설계

정방향/역방향 이산웨이블릿 변환은 같은 컨볼루션 필터 연산을 하나, 필터 계수가 다르고 다운/업 샘플링 과정만이 차이를 갖는다. 따라서 그림 6의 정방향/역방향 이산웨이블릿 변환기는 컨볼루션 연산 처리를 위한 곱셈&저장과 검색&덧셈의 2단 파이프라인 통합 아키텍처를 보인다.

다른 필터계수 때문에 정방향 LUT/역방향 LUT를 별도로 구성한 곱셈부를 가진다. 각각의 LUT는 고대역 및 저대역에 따라 LMW 병렬화 곱셈값을 다르게 구현

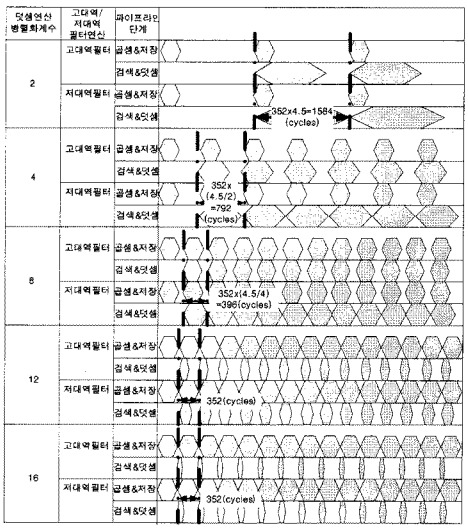


그림 4. DWT에서 덧셈연산 병렬화에 따른 파이프라인 동작비교

Fig. 4. Comparative pipelined operations on the parallelism of addition for DWT.

표 1. 덧셈연산 병렬화에 따른 파이프라인 주기 및 실행효율

Table 1. Pipeline stage and efficiency on the parallelism of addition.

덧셈연산 병렬화 계수	파이프라인 주기	파이프라인 실행효율
2	1,584	0.61
4	792	0.72
8	396	0.94
12	352	0.86
16	352	0.78

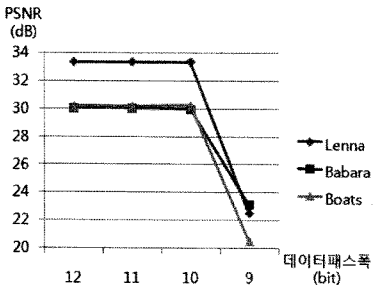


그림 5. 영상의 PSNR vs. 데이터패스폭

Fig. 5. Image PSNR vs. datapath bitwidth.

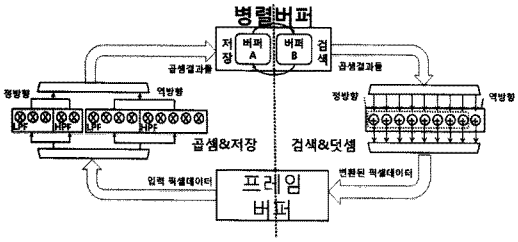


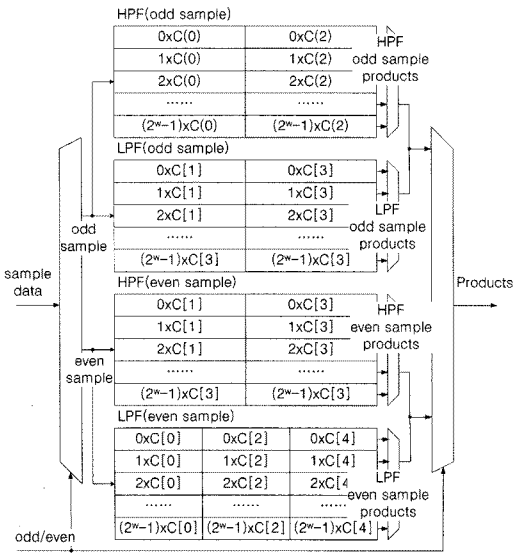
그림 6. 정방향/역방향 이산웨이블릿 변환기 통합 아키텍처

Fig. 6. Unified architecture of DWT/IDWT.

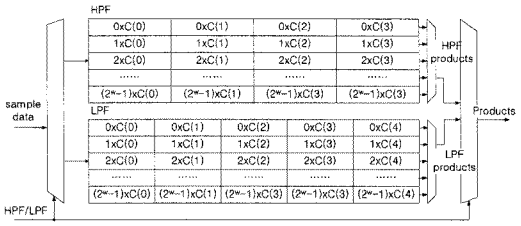
하고 있는데, 정방향 LUT는 저(고) 대역에 3(2) LMW 병렬화 곱셈값을 역방향 LUT는 저(고)대역에 4(5) LMW 병렬화 곱셈값을 처리한다. 정방향 이산웨이블릿 변환에서 곱셈부의 곱셈연산은 픽셀데이터 당 1회의 곱셈연산을 갖는 것에 비해, 덧셈부는 픽셀데이터 당 필터탭 수(9+7)×1/2= 8회의 덧셈연산을 갖는다. 덧셈부에 8개의 병렬가산기를 이용하여 덧셈연산을 수행하면 곱셈부와 덧셈부의 연산 균형을 맞출 수 있으며 효율적 파이프라인 동작이 가능하다. 역방향 이산웨이블릿 변환에서의 덧셈 연산과정은 고대역/저대역 필터 연산 외에 필터연산된 결과를 더하는 덧셈연산이 추가로 요구된다. 따라서 역방향 이산웨이블릿 변환에서 덧셈부는 픽셀데이터 당 9회의 덧셈연산을 갖는다. 역방향에서 덧셈부는 9개의 병렬가산기를 사용하여 덧셈연산을 수행하면 효율적 파이프라인 동작이 가능하다. 통합된 정방향/역방향 이산웨이블릿 변환기에서 덧셈부는 검색 및 덧셈의 파이프라인 기본동작이 같기 때문에 그림 6과 같이 파이프라인을 위한 버퍼와 덧셈기를 공유할 수 있다.

2. 병렬 곱셈기 설계

정방향/역방향 이산웨이블릿을 위한 병렬 곱셈기는 LUT 기반 DA 구조의 LMW(Long-Multiplicand-Word) 곱셈기로 구성된다. 정방향 이산웨이블릿 변환에서 다운샘플링 과정을 살펴보면 홀수/짝수 픽셀데이터에 대해 각각 다른 필터계수를 곱한다. 따라서 정방향 이산



(a) 정방향 이산웨이블릿 곱셈기 구조



(b) 역방향 이산웨이블릿 곱셈기 구조

그림 7. 병렬 LMW-LUT 곱셈기의 구조

Fig. 7. Parallel LMW-LUT multiplier architecture.

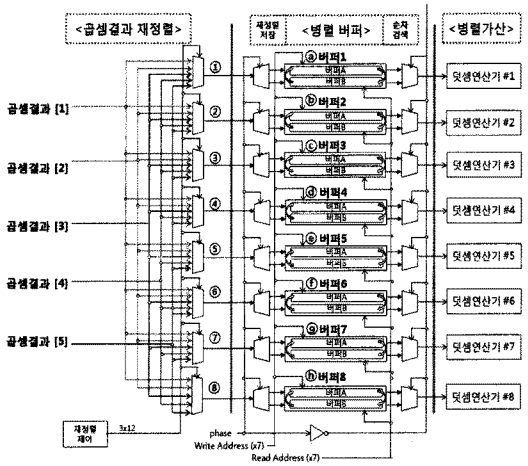
웨이블릿 변환을 위한 병렬곱셈기는 그림 7(a)와 같이 홀수/짝수 픽셀데이터에 대한 고대역/저대역 필터 LUT로 나뉘어 구성된다. 입력 픽셀데이터가 홀수번째 데이터이면 고대역 필터 LUT는 필터계수 {C(0), C(2)}와 병렬곱셈한 결과를 출력하고, 저대역 필터 LUT는 필터계수 {C(1), C(3)}와 병렬곱셈한 결과를 출력한다. 입력 픽셀데이터가 짝수번째 데이터이면 고대역 필터 LUT에서 {C(1), C(3)}와 병렬곱셈한 결과를 출력하고, 저대역 필터 LUT에서 {C(0), C(2), C(4)}와 병렬곱셈한 결과를 출력하여 컨볼루션 곱셈연산을 처리한다.

역방향 이산웨이블릿 변환은 정방향 이산웨이블릿 변환과 달리 홀수/짝수 픽셀데이터에 대한 처리과정이 존재하지 않는다. 반면에 역방향 이산웨이블릿 변환을 위한 병렬곱셈기는 그림 7(b)와 같이 고대역/저대역 필터 LUT로 나뉘어 구성된다. 픽셀 데이터는 고대역/저대역 필터 LUT의 주소로 병렬 입력되고 {C(0), C(1), C(2), C(3)}/{C(0), C(1), C(2), C(3), C(4)}와의 병렬곱셈 결과를 출력하여 컨볼루션 곱셈연산을 수행한다.

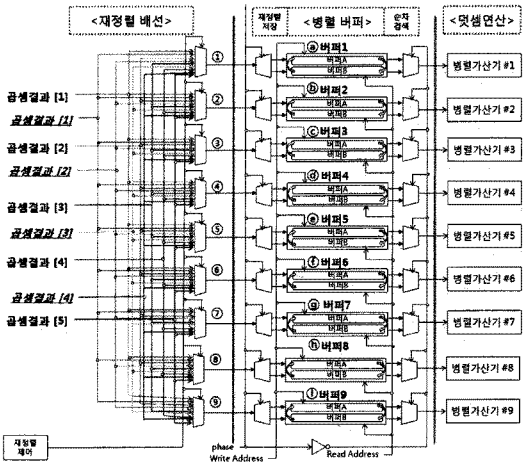
3. 버퍼 재정렬

곱셈연산을 고속 병렬화시키고, 대응된 덧셈연산을 병렬화시켰으나, DA를 이용한 LUT 기반의 곱셈연산은 덧셈연산에 비해 연산 속도가 더 빠르다. 따라서 이산웨이블릿 변환기의 곱셈&저장, 검색&덧셈 2단 파이프라인 단계가 균형을 이루도록 덧셈연산에 필요한 입력데이터를 고속 순차검색할 수 있게 버퍼를 설계하였다. 정방향/역방향 이산웨이블릿 변환기에서 병렬곱셈 결과를 버퍼로 저장하기 위한 구조는 그림 8과 같이 곱셈결과 <재정렬 배선>, <병렬버퍼>블럭으로 구성하였다. <병렬버퍼>블럭은 입력버퍼(버퍼A)와 검색버퍼(버퍼B)로 구성되어 인터리빙 동작한다. 곱셈결과 [1]~[5]를 병렬입력버퍼[A]에 <재정렬 배선>블록을 통해 배선(routing) 저장시키는 동안 각각의 곱셈결과값들은 검색버퍼[B]로부터 고속 순차검색되어 덧셈부의 병렬가산기에서 누적가산된다. 라인단위의 필터연산이 끝나면 병렬입력버퍼[A]와 검색버퍼[B]의 역할 및 기능이 바뀌면서 인터리브(interleaved) 버퍼링이 이루어진다. 이러한 곱셈부와 덧셈부 사이의 재정렬 버퍼구조는 고속 곱셈연산과 연산속도가 느린 덧셈연산간의 연산속도 차이를 감소시킴으로써, 컨볼루션 파이프라인의 효율을 높이고 이산웨이블릿 변환기의 연산처리성능을 증가시킨다.

곱셈결과를 재정렬하여 병렬 저장하고 병렬덧셈부의



(a) 정방향 필터 연산



(b) 역방향 필터 연산

그림 8. 정방향/역방향 및 고대역/저대역 필터의 재정렬 버퍼 구조

Fig. 8. Alignment buffer architecture for DWT/IDWT & HPF/LPF.

가산기들이 고속 순차검색할 수 있도록 정렬하는 주소를 할당하는 알고리즘은 그림 8(a)의 정방향 필터연산에 대해서 표 2와 같이 설계된다. 먼저 1 라인의 352개 픽셀데이터는 p(0)부터 p(351)까지 순차적으로 입력된다. 그리고 픽셀데이터와 필터계수들과의 곱셈결과는 대칭성과 다운샘플링에 의해 곱셈결과[1]~[5]로 생성된다. 실제로 픽셀데이터 p(0)의 곱셈결과[2] p(0)*c(1)은 컨볼루션 연산 결과 데이터 p'(1)과 p'(351)의 누적연산에 필요하므로 버퍼 ①과 버퍼 ④에 저장된다. 그리고 픽셀데이터 p(0)의 곱셈결과[4] p(0)*c(3)는 필터출력값

표 2. 정방향 고대역 필터 연산을 위한 재정렬 배선 알고리즘

Table 2. Alignment routing algorithm for DWT&HPF computation.

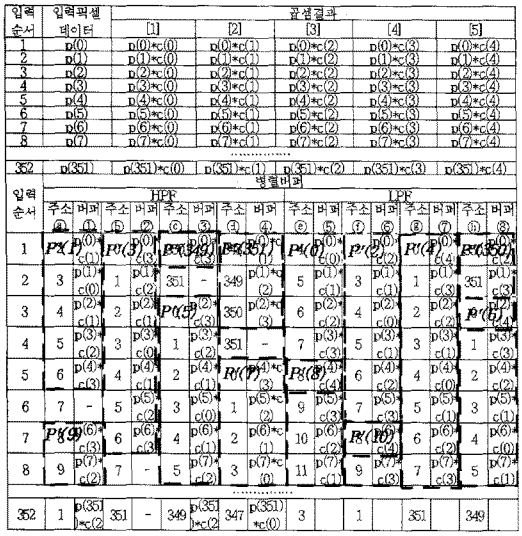
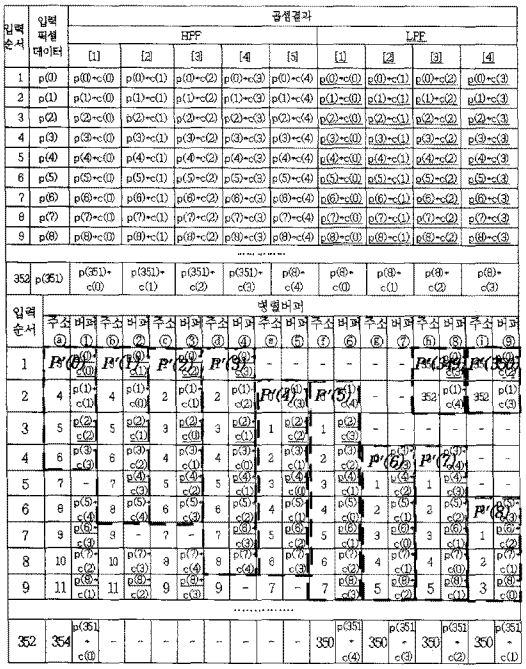


표 3. 역방향 필터 연산을 위한 재정렬 배선 알고리즘

Table 3. Alignment routing algorithm for IDWT computation.



p′(3)와 p′(351)의 누적연산에 필요하므로 버퍼2와 버퍼3에 저장된다. 이같은 방법으로 재정렬된 곱셈결과 [1]~[5]는 버퍼①~⑧에 배선 저장되는데, 곱셈결과 [1]~[5]의 배선(routing)주소는 테이블의 주소 ⓐ~ⓗ와 같이 할당하여 필터출력 데이터를 위한 덧셈연산 과정에 순차검색이 가능하도록 한다. 곱셈결과들의 배선 저장규칙은 8픽셀마다 반복되면서, 병렬 컨볼루션 곱셈결과 [1]~[5]를 병렬 가산기#1~#8에 매핑시키는 기능이 그림 8(a) <재정렬 배선>블럭과 같이 구현된다.

정방향 이산웨이블릿 변환에서는 고대역 필터 연산된 데이터과 저대역 필터 연산된 데이터로 나누어 계산해서 저장하였으나, 역방향 이산웨이블릿 연산에서는 고대역 필터 연산된 데이터와 저대역 필터 연산된 데이터를 합산한다. 따라서 그림 8(b)와 같이 재정렬 배선 블록에서 고대역 필터 연산된 데이터(곱셈결과 [1]~[5])와 저대역 필터 연산된 데이터(밑줄친 곱셈결과 [1]~[4])를 병렬버퍼에 순차 저장하도록 저장알고리즘을 표 3과 같이 설계하였다. 병렬버퍼에 저장되는 배선(routing)주소는 테이블의 주소 ⓐ~ⓘ와 같이 할당하여 필터출력 데이터를 위한 덧셈연산과정에 순차검색이 가능하도록 하였으며, 곱셈결과들의 배선 저장규칙은 9픽셀마다 반복되도록 그림 8(b)의 <재정렬 배선>블럭과 같이 설계하였다.

4. 병렬 가산기 설계

병렬가산기 블록은 그림 9와 같이 병렬버퍼에 저장된 곱셈결과를 순차검색하여 필터탭 수 만큼 누적가산기로 연산하여 컨볼루션된 데이터를 생성한다. 정방향 웨이블릿 변환에서 컨볼루션된 데이터들은 그림 10과

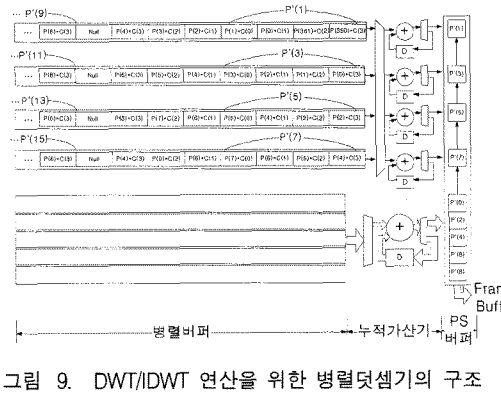


그림 9. DWT/IDWT 연산을 위한 병렬덧셈기의 구조

Fig. 9. Parallel accumulator architecture for DWT/IDWT computation.

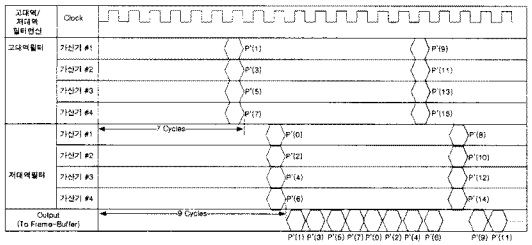


그림 10. 정방향 컨볼루션된 데이터 생성
Fig. 10. DWT data output of forward convolution.

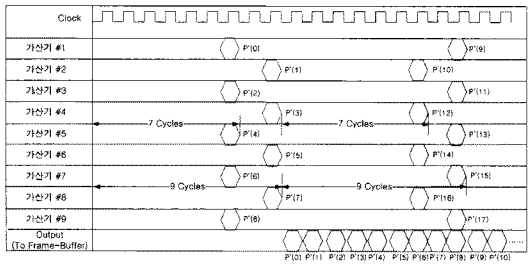


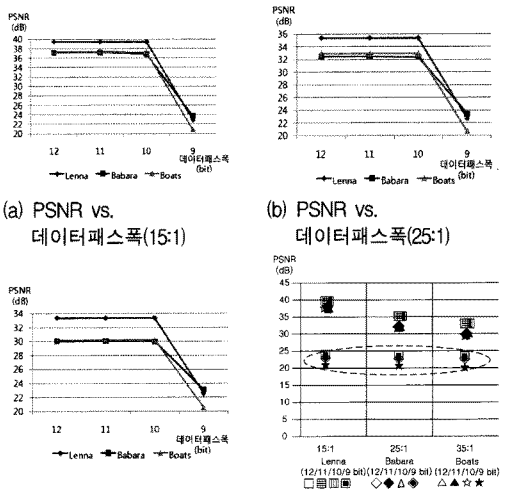
그림 11. 역방향 컨볼루션된 데이터 생성
Fig. 11. IDWT data output of reverse convolution.

같이 고대역에서 7 사이클마다 4개의 데이터가 생성되며, 저대역에서 9 사이클마다 4개의 데이터가 생성된다. 컨볼루션된 데이터들은 1개씩 프레임버퍼에 저장되는데, Parallel-to-Serial 버퍼는 병렬 생성된 데이터들을 저장하였다가 다음 누적연산 시 한 개씩 프레임버퍼로 데이터를 전송한다. 역방향 웨이블릿 변환에서는 병렬버퍼에 순차적으로 저장된 고대역 필터 연산된 데이터와 저대역 필터 연산된 데이터를 검색, 누적 연산하여 컨볼루션된 데이터들을 구한다. 그림 11과 같이 컨볼루션된 데이터들은 홀수픽셀 데이터의 경우 7사이클 마다 생성되고, 짝수픽셀 데이터의 경우 9사이클마다 생성된다.

IV. 실험 및 고찰

1. 화질성능 최적화 데이터패스

JPEG2000의 압축률과 이산웨이블릿 변환기의 데이터패스폭에 따른 JPEG2000의 화질을 비교하기 위해 압축률(15:1/25:1/35:1)및 데이터 패스폭(9/10/11/12비트)에 대하여 DWT/IDWT를 실험하였다. 그림 12는 JPEG2000의 압축률과 이산웨이블릿 변환의 데이터패스폭에 대해 DWT/IDWT의 Babara, Lena, Boats 영상 PSNR을 비교한 그래프이다. 그림 12(d)에서 30dB이상



(a) PSNR vs. 데이터패스폭(15:1)
(b) PSNR vs. 데이터패스폭(25:1)
(c) PSNR vs. 데이터패스폭(35:1)
(d) PSNR과 데이터패스폭, 압축률

그림 12. JPEG2000의 압축률과 웨이블릿 변환기의 데이터패스폭에 따른 영상의 PSNR 비교
Fig. 12. PSNR comparison for datapath bitwidth & compression rate.

고화질의 영상복원을 보이는 그룹(실선 타원으로 표시)과 20dB 근처의 화질저하가 발생하는 그룹(점선 타원으로 표시)으로 구분된다. 20dB로 화질저하가 발생하는 그룹들은 이산웨이블릿 변환의 데이터패스폭이 9비트인 DWT/IDWT 영상들이다. 압축률 변화에 따른 부/복호화된 영상의 PSNR을 살펴보면, 압축률이 15:1에서 35:1로 변함에 따라 화질 저하는 많이 발생하지 않음을 알 수 있다. 따라서 JPEG2000 코덱은 압축률 변화에 따른 화질저하보다는 이산웨이블릿 변환기의 데이터패스폭에 따른 화질저하가 크게 발생하는 특징이 확인되고 있다.

2. 정방향/역방향 이산웨이블릿 변환기의 데이터 처리성능

정방향/역방향 이산웨이블릿 변환기의 하드웨어 성능을 확인하기 위해 Verilog-HDL로 설계하여 Altera/Xilinx FPGA와 ASIC(0.18um) 공정으로 시뮬레이션하였다. 표 4는 설계 구현된 이산웨이블릿 변환기를 기존 연구결과들과 크기를 비교한 결과이다. 정방향/역방향 이산웨이블릿 변환기를 통합 설계한 결과, 기존 연구[5]에 비해 게이트 수는 1/2인 작은 반면에 메모리는 2.7배로 많이 요구하고 있다. 메모리는 정방향/역방향 이산웨이블릿 변환기 내 LMW-LUT 기반 곱셈부와

표 4. 이산웨이블릿 변환기의 크기 비교
Table 4. Implementation results for DWT/IDWT.

기존연구	Transform	공정	크기	메모리(Kbits)
이경민 외 1인[5]	F	Xilinx Virtex II	27,851 gates	256
윤기태 외 1인[6]	F	ASIC 0.18 um	118,768 gates	64
Silva S.V. et al[7]	F	Altera Stratix II	785 LEs	N/A
Masud, S. et al[8]	F	Altera Stratix II	480 LEs	N/A
Longa, P. et al[9]	F	Altera Stratix II	495 LEs	N/A
Longa, P. et al[10]	F	Altera Stratix II	422 LEs	N/A
본 연구	F&I	Xilinx Virtex II	2,456 Slice LUTs	1,010
		Altera Stratix II	630 LEs	1,010
		ASIC 0.18 um	14,800 gates	1,010

표 5. 정방향/역방향 이산웨이블릿 변환기의 실시간 처리 성능 비교
Table 5. Performance comparisons of DWT/IDWT implementation.

기존연구	Resolution	공정	데이터처리성능 (frames/s)	동작주파수(MHz)
이경민 외 1인[5]	640x480	Xilinx virtex II	10	50
본 연구	640x480	Xilinx virtex II	30	58
	800x600	0.18um	30	90

표 6. 정방향/역방향 이산웨이블릿 변환기의 데이터 처리 성능 비교
Table 6. Throughput comparisons of DWT/IDWT implementation.

기존연구	공정	데이터처리성능 (Msamples/s)
Silva S.V. et al[7]	Altera Stratix II	88.5
Masud, S. et al[8]	Altera Stratix II	44
Longa, P. et al[9]	Altera Stratix II	144.9
Longa, P. et al[10]	Altera Stratix II	75.5
본 연구	Xilinx Virtex II	98
	Altera Stratix II	108.7
	0.18um	130

재정렬 버퍼를 설계하였기 때문에 요구되고 있다. 본 연구와 기존 연구(구현된 면적의 2배)의 정방향/역방향 통합 이산웨이블릿 변환기 크기를 비교하면, 가장 작게 구현된 연구[10]에 비해 25%만큼 개선된 결과를 보이고 있다.

설계된 정방향/역방향 이산웨이블릿 변환기의 동영상에 대한 실시간 처리 성능을 비교하면 표 5와 같다. FPGA 시뮬레이션하였을 때 VGA 동영상(640x480)에 대해 동작주파수 58MHz에서 초당 30프레임으로 실시간 처리하며, 기존 연구[5]에 비해 2.6배의 향상된 데이터 처리성능을 보여준다. ASIC 시뮬레이션한 결과는 90MHz에서 SVGA 동영상(800x600)을 30프레임 실시간으로 처리 가능함을 보였다.

표 6과 같이 이산웨이블릿 변환기의 데이터 처리 성능을 기존 연구와 비교하였다. 정방향/역방향 이산웨이블릿 변환기를 설계한 본 연구를 Xilinx FPGA(Virtex II) 시뮬레이션 한 결과 초당 98×10^6개의 샘플데이터를 처리하고, Altera FPGA(Stratix II)에서 초당 108.7×10^6개의 샘플데이터를 처리하며, ASIC(0.18um)에서는 130×10^6/ sec개의 샘플데이터를 변환한다. 표 6을 살펴보면 컨볼루션 기반의 구조[7, 9~10]가 리프팅 기반의 구조[8]에 비해 고속 연산처리함을 알 수 있다. non-cascaded 구조인 본 연구와 cascaded 구조[7, 9~10]인 기존 연구를 비교하면, non-cascaded 구조가 고속의 cascaded 구조에 비해 거의 근접한 성능을 내면서 데이터 처리 성능이 개선됨을 보이고 있다.

V. 결 론

본 연구에서는 정방향/역방향 이산웨이블릿 변환의 컨볼루션/non-cascaded/병렬 구조를 2단 파이프라인 구조로 설계하여 효과적인 최적의 고속 연산구조를 제안하고 있다. 컨볼루션 필터의 대칭성과 업/다운 샘플링을 통해서 컨볼루션의 곱셈연산을 1/4로 감소시켰으며, LMW(Long Multiplicand Word)-LUT 기반의 병렬 곱셈기 구조를 설계해서 곱셈연산 성능을 2.5/4.5배 개선시킬 수 있었다. 이산웨이블릿 변환기의 컨볼루션 연산을 곱셈&저장과 검색&덧셈의 2단 파이프라인 데이터패스로 병행처리하고, 컨볼루션 누적덧셈기의 8병렬화 및 중간버퍼의 재정렬기능을 통해서 파이프라인 단계의 효율을 높일 수 있었다. 또한 화질성능의 최적화를 위한 데이터패스폭을 구하기 위해서 정방향/역방향 이산웨이블릿 변환 연산과정의 양자화 오류와 데이터패스폭 간의 관계를 시뮬레이션해서 최적의 10비트 데이터패스폭을 설계하였다. 정방향/역방향의 이산웨이블릿변환기를 Xilinx FPGA와 ASIC(0.18um)공정에서 시뮬레이션하여 로직은 1/2이상 작아지고 메모리는 2.7배 가량 증가됨을 확인하였으며, FPGA 환경에서 58MHz에서 VGA 동영상을 30fps 실시간 처리하고 ASIC 환경에서 SVGA 동영상을 30fps 실시간 처리함을 보였다.

참 고 문 헌

[1] H. Liao, M. K. Mandal, and B. F. Cockburn, "Efficient Architecture for 1-D and 2-D Lifting-based Wavelet Transform," IEEE transactions Signal Processing., vol. 52, no.5, pp. 1315-1326, May 2004.

[2] F. Marino, D. Guevorkian, and J. Astola, "Highly efficient high-speed/low-power architectures for 1-D discrete wavelet transform," IEEE Trans. CAS-II, vol. 47, no. 12, pp.1492-1502, December 2000.

[3] K. Parhi, T. Denk, "Systolic VLSI architectures for 1-D wavelet transform," in Proc. 32nd IEEE Asilomar Conference on Signals, Systems and Computers, Pacific Grove, Canada, vol. 2, pp.1220-1224, November 1998.

[4] C. Chakrabati, M. Vishwanath, and R. M. Owens, "Architecture for wavelet transforms:a survey," J. VLSI Signal Process, vol. 14, no. 2, pp. 171-192, February 1996.

[5] 이경민, 김영민, "JPEG2000 CODEC을 위한 DWT 및 양자화기 VLSI 설계," 전자공학회논문지, 제40권 SD편, 제1호, 45-51쪽, 2003년 1월

[6] 윤기태, 최준림, "JPEG2000을 위한 리프팅 방식의 DWT 필터 하드웨어 설계", 2008년 SoC학술대회, 375-378쪽, 평창, 대한민국, 2008년 6월

[7] Masud, S., and McCanny, J., "Reusable silicon IP cores for discrete wavelet transform application" IEEE Trans. Circuits Syst. I, Fundam. Theory, pp. 1114 - 1124, April 2004.

[8] Silva, S.V., and Bampi, S., "Area and throughput trade-offs in the design of pipelined discrete wavelet transform architectures", Proc. Design, Automat. and Test in Europe Conf. Exhibition (DATE'05), 2005, Vol. 3, pp. 32 - 7

[9] Longa, P., Miri, A., and Bolic, M., "A flexible design of filterbank architectures for discrete wavelet transforms". Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP'07), Vol. 3, pp. 1441 - 444, April 2007.

[10] Longa, P., Miri, A., and Bolic, M., "Modified distributed arithmetic based architecture for discrete wavelet transforms," Electronics Letters Volume 44, Issue 4, pp. 270 - 271 , February 2008.

저 자 소 개

이 승 권(정회원)

2002년 광운대학교 컴퓨터공학과 학사 졸업.

2004년 광운대학교 컴퓨터공학과 석사 졸업.

2004년~현재 광운대학교 컴퓨터공학과 박사과정

2004년~현재 (주)동운아나텍 연구개발본부 시스템팀 팀장

<주관심분야 : 영상신호처리, SoC설계, 임베디드 시스템>

공 진 흥(평생회원)

1980년 서울대학교 전자공학과 학사 졸업.

1982년 한국과학기술원 전기 및 전자공학과 석사 졸업.

1989년 텍사스주립대학교 컴퓨터공학과 박사 졸업.

<주관심분야 : 영상신호처리, SoC설계, 임베디드 시스템>