2013년 5월 전자공학회 논문지 제 50 권 제 5 호
Journal of The Institute of Electronics Engineers of Korea Vol. 50, NO. 5, May 2013 http://dx.doi.org/10.5573/ieek.2013.50.5.102

논문 2012-50-5-12

4K UHD급 H.264/AVC 복호화기를 위한 4×4 블록 병렬 보간 움직임보상기 아키텍처 설계

(A Design of 4×4 Block Parallel Interpolation Motion Compensation Architecture for 4K UHD H.264/AVC Decoder)

이 경 호*, 공 진 흥**

(Kyung-Ho Lee© and Jin-Hyeung Kong)

요 약

본 연구에서는 4K UHD(3840×2160) 영상을 실시간 복호화하기 위한 4×4 블록 병렬 보간 H.264/AVC 움직임보상기를 제안한다. 연산처리 성능을 향상시키기 위해 보간 연산을 4×4 블록 단위로 병렬화시켰으며, 병렬 보간 연산에서 필요한 메모리 대역폭을 확장하기 위해 9×9개의 메모리 어레이를 가진 2D 캐쉬 버퍼 구조를 설계하였다. 그리고 2D 캐쉬 버퍼는 검색영역 간 재사용 기법을 적용하여 참조화소의 중복저장을 최소화하였으며, 4×4 블록 병렬 보간 필터는 3단(수평·수직 1/2부화소, 대각선 1/2부화소, 1/4부화소) 평면 보간 연산 파이프라인 구조로 설계하여 연산회로를 고속화시켰다. 0.13um 공정에서 시뮬레이션한 결과, 161K게이트의 H.264/AVC 움직임보상기는 동작주파수 150MHz에서 4K UHD급 동영상을 초당 72프레임으로 실시간 처리하는 성능을 보였다.

Abstract

In this paper, we proposed a 4×4 block parallel architecture of interpolation for high-performance H.264/AVC Motion Compensation in 4K UHD(3840×2160) video real time processing. To improve throughput, we design 4×4 block parallel interpolation. For supplying the 9×9 reference data for interpolation, we design 2D cache buffer which consists of the 9×9 memory arrays. We minimize redundant storage of the reference pixel by applying the Search Area Stripe Reuse scheme(SASR), and implement high-speed plane interpolator with 3-stage pipeline(Horizontal·Vertical 1/2 interpolation, Diagonal 1/2 interpolation, 1/4 interpolation). The proposed architecture was simulated in 0.13um standard cell library. The maximum operation frequency is 150MHz. The gate count is 161Kgates. The proposed H.264/AVC Motion Compensation can support 4K UHD at 72 frames per second by running at 150MHz.

Keywords : H.264/AVC Decoder, Motion Compensation, 4×4 Block Parallel, 2D Cache buffer, Pipeline

Ⅰ. 서 론

현재 FHD(Full High Definition)를 넘어 UHD(Ultra High Definition)급 초고해상도의 영상 장치들이 시장에 등장하고 있다. 이러한 초고해상도 영상 장치들은 많은 연산량과 높은 데이터 대역폭의 실시간 처리를 요구하여 뛰어난 압축성능을 가진 영상 압축기술을 필요로 한다. 최신 영상압축 기술인 H.264/AVC는 기존 영상압축 기술(MPEG-2, MPEG-4, H.263)에 비해 약 39%~64% bit-rate를 절약할 수 있는 뛰어난 압축 성능을 가진다[1]. H.264/AVC 복호화기의 움직임보상은 시간적 중복성을 복원하는 기술로써 부호화 효율을 높이기 위해 가변블록크기와 다양한 참조 프레임, 1/4 화소의 정밀 보간을 지원한다. 다양한 참조 프레임과 가변블록크기를 적용하기 위해 메모리 검색량이 늘어나 움직임보상이

* 정회원, ** 평생회원, 광운대학교 컴퓨터공학과
(Department of Computer Engineering, Kwangwoon University)
※ 본 연구는 지식경제부가 지원하는 산업융합원천기술개발사업을 통해 개발된 결과임을 밝힙니다. (10039173 융복합 혁신반도체 기술개발)
© Corresponding Author(E-mail:tyvex@naver.com)
접수일자: 2013년1월4일, 수정완료일: 2013년4월24일

H.264/AVC 복호화기 전체 메모리 검색량의 약 75%를 차지한다[2]. 또한 1/4화소 정밀 보간 연산으로 인해 움직임보상의 연산량은 H.264/AVC 복호화기 전체 연산량의 약 35%를 차지한다[3]. 따라서 연산량 및 메모리 검색량을 해결하기 위하여 보간 연산 병렬화를 통한 보간 연산 고속화와 데이터 재사용을 통한 외부메모리 검색 감소에 관한 연구가 병행 진행되고 있다[4~10].

보간 연산 고속화 연구는 6-Tap FIR 필터를 기본 연산 단위로 병렬화하여 연산처리 성능을 높였다. 4개의 6-Tap FIR 필터를 병렬화한 연구[4, 10] (1-D (Dimensional) FIR 보간 필터)는 4개의 1/2부화소 또는 1/4부화소를 한 번에 생성한다. 4×4 보간블록을 생성하기 위해 움직임 벡터가 정수화소나 수평·수직 1/2부화소일 때에는 4회의 보간 연산을 한다. 그러나 움직임 벡터가 대각선 방향의 1/2부화소일 때에는 수평 1/2부화소를 보간하고 대각선 1/2부화소를 구하여 총 9회의 보간 연산을 한다. 따라서 4개의 6-Tap FIR 필터를 병렬화한 연구는 움직임 벡터에 따라 연산 횟수가 달라 가변적인 제어를 해야 하고, 대각선 1/2부화소 보간에서 늘어난 연산횟수 때문에 실시간 처리는 저해상도 영상으로 제한된다. 가변적인 제어 복잡도를 해결하고 대각선 1/4부화소의 연산횟수를 단축시킨 연구[5~6](2-D FIR 보간 필터)는 수평·대각선 1/2부화소 보간 및 1/4부화소 보간의 데이터 종속을 파이프라인 구조로 병행처리하여 1개 보간 화소를 구하는 연산의 횟수를 최소화하였다. 그러나 1개 보간 화소 생성을 위한 참조 화소의 입력지연시간이 길고 4×4 블록 보간 화소 생성 시 인접 보간 화소 간 재사용이 없어 중복연산 된다. 이 같은 입력지연시간과 중복연산을 줄이기 위해서 연구[7~8](Separate 1-D FIR 보간 필터)는 수평 1/2부화소를 생성하는 1-D FIR 보간 필터와 수직과 대각선 1/2부화소를 생성하는 1-D FIR 보간 필터가 각각 병렬 연산한다. 수평 1/2부화소를 생성할 때 사용한 참조 화소를 수직·대각선 1/2부화소를 생성할 때 재사용하여 중복연산 및 입력지연시간을 감소시켰으며, 또한 수평 1/2부화소 생성 결과를 대각선 1/2부화소 생성에 재사용하였다. 그러나 Separate 1-D FIR 보간 필터구조는 4×1라인단위로 보간 화소를 생성하는 병렬성의 제한 때문에 고해상도(FHD) 영상의 실시간 처리까지만 가능한 연산처리 성능을 갖는다. 이 같은 성능을 개선하고자 Separate 1-D FIR 보간 필터구조 2개를 이용한 연구[9]는 인접한 4×4 블록 2개 내에서 4×1라인단위의 보간 연산을 처리하여 기존 Separate 1-D FIR 보간 필터구조[7~8]에 비해 연산처리성능을 2배 증가시킬 수 있었다. 그러나 인접한 4×4 블록의 참조화소들을 저장, 관리하고 연산결과를 동시에 제어하기 위한 연산회로의 복잡도로 인해 Separate 1-D FIR 보간 필터구조에 비해 면적이 2.7배로 늘어났으며, 4K UHD(3840×2160)를 실시간 처리하는데 동작주파수가 2배로 증가하여 전력소모가 커지는 문제점을 보이고 있다.

본 연구에서는 4K UHD 영상에 대한 H.264/AVC 움직임보상을 실시간 처리하기 위해 4×4 블록 병렬 보간 필터 구조와 2D 캐쉬 버퍼 구조를 제안한다. 제안하는 4×4 블록 병렬 보간 필터는 4×1라인단위가 아닌 4×4 블록단위로 보간 화소를 생성할 수 있도록 부화소를 동시에 보간하는 평면 보간 연산구조로 설계하였다. 평면 보간 연산구조는 수평 및 수직 1/2부화소를 생성하는 1단 평면 FIR 필터, 대각선 1/2부화소를 생성하는 2단 평면 FIR 필터, 그리고 마지막 1/4부화소를 생성하는 3단 평면 Bilinear 필터의 3단 파이프라인 구조로 연산처리성능을 향상시켰다. 그리고 4×4 블록단위 보간 연산구조에서 요구되는 참조 화소를 실시간으로 공급하기 위해 2D 캐쉬 버퍼 구조로 설계했고, 움직임 벡터에 따른 중복적인 참조메모리 검색을 최소화하기 위해 검색영역 간 재사용 기법을 사용하였다.

본 논문의 구성은 Ⅱ장에서 H.264/AVC 움직임보상의 4×4 블록 병렬 보간 연산 알고리즘을 설명하고, Ⅲ장에서는 제안하는 4×4 블록 병렬 보간 움직임보상기 아키텍처에 대해 설계한다. Ⅳ장에서는 새로운 아키텍처의 성능을 다른 연구결과와 비교하며, 마지막으로 Ⅴ장에서는 결론을 맺는다.

Ⅱ. H.264/AVC 움직임보상 연산의 4×4 블록 병렬 보간

H.264/AVC 움직임보상은 움직임 벡터가 가리키는 블록의 참조 화소들을 참조프레임으로 부터 가져와 현재 영상을 복원한다. 움직임보상의 보간 연산은 그림 1과 같이 1/2부화소와 1/4부화소 보간을 수행한다. 수직 1/2부화소(h, m)보간은 참조프레임의 정수화소({A, C, G, M, R, T}, {B, D, H, N, S, U})를 6-Tap FIR 필터 연산으로 생성한다. 수평 1/2부화소(b, s)보간은 참조프레임의 정수화소({E, F, G, H, I, J}, {K, L, M, N, P, Q})를 6-Tap FIR 필터 연산을 통해 얻는다. 대각선 1/2부화소(j)의 경우 6-Tap FIR 필터 연산으로 계산된 수평 1/2부화소({aa, bb, b, s, gg, hh})또는 수직 1/2부

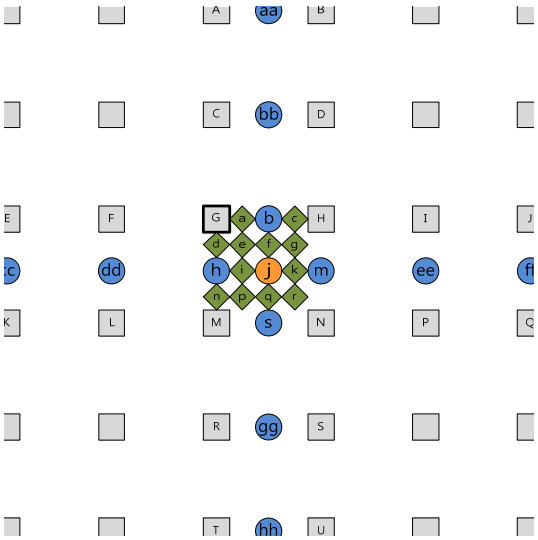


그림 1. 움직임보상의 보간 연산
Fig. 1. Interpolation of Motion Compensation.

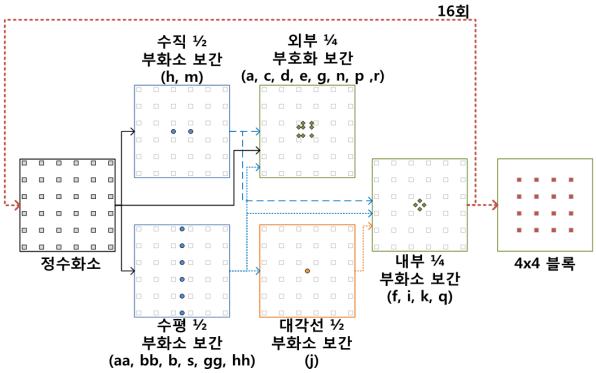


그림 2. 보간 연산 과정의 데이터 종속성
Fig. 2. Data dependancy of Interpolation procedure.

화소({cc, dd, h, m, ee, ff})를 6-Tap FIR 필터 연산하여 생성한다. 1/4부화소 보간은 외부 1/4부화소(a, c, d, n, e, g, p, r)보간과 내부 1/4부화소(f, i, k, q)보간으로 나뉜다. 외부 1/4부화소 보간은 정수화소와 수평 1/2부화소간 또는 정수화소와 수직 1/2부화소간 또는 수평 1/2부화소와 수직 1/2부화소간 bilinear연산으로 생성한다. 반면 내부 1/4부화소 보간은 수평 및 수직 1/2부화소와 대각선 1/2부화소간 bilinear연산으로 생성한다. 따라서 1/2부화소 및 1/4부화소 보간 연산은 그림 2와 같은 데이터 종속성을 가진다. 그러므로 보간 화소 16개를 생성하기 위해서는 그림 2의 순환 연산과정을 16회 반복 수행해야 한다.

부화소 보간 연산 단위를 기존 1개 정수화소(G)의 인접 부화소에서 1×4 라인 화소의 인접 부화소로 병렬 확장한 Separate 1-D FIR 보간 연산은 그림 3과 같이 나타난다. 그림 3 (a)와 같이 수평 1/2 부화소 보간 과정

(a) 수평 1/2부화소 병렬보간

(b) 수직 1/2부화소 병렬보간

(c) 대각선 1/2부화소 병렬보간

(d) 1/4부화소 병렬보간

그림 3. Separate 1-D FIR 병렬보간 연산
Fig. 3. Parallel Interpolation of Separate 1-D FIR.

에서는 1×4개의 6-Tap FIR 필터가 입력된 1×9 정수화소를 공유, 참조하여 수평 1/2 부화소 1×4개를 생성한다. 그리고 그림 3 (b)와 같이 수직 1/2 부화소 보간 과정에서는 1×5개의 6-Tap FIR 필터가 수평 1/2 부화소 보간에서 사용한 6×9 정수화소를 재사용하여 수직 1/2 부화소 1×5개를 생성한다. 그리고 그림 3 (c)와 같이 대각선 1/2부화소 보간 과정에서는 1×4개의 6-Tap FIR 필터가 수직 1/2 부화소 보간에서 생성한 6×4개의 1/2 부화소를 재사용하여 대각선 1/2 부화소 1×4개를 생성

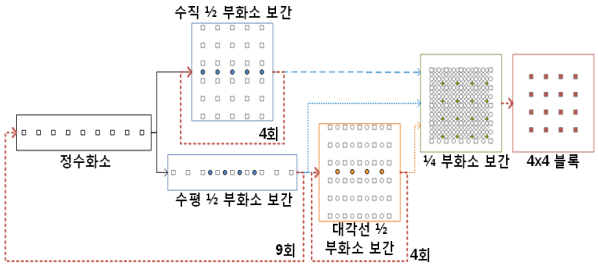
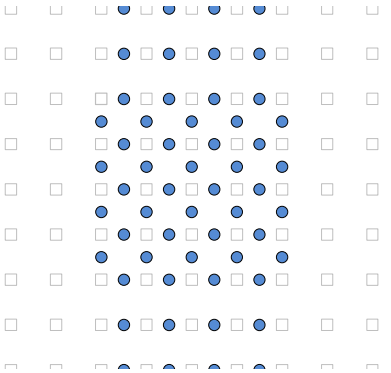


그림 4. Separate 1-D FIR 병렬보간 연산과정
Fig. 4. Process of Separate 1-D FIR parallel Interpolation.

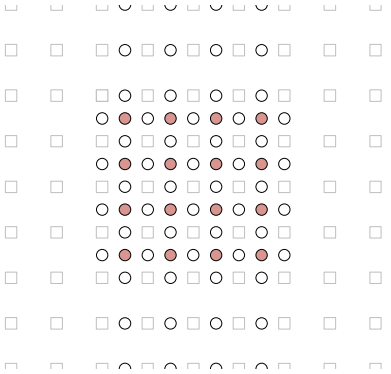
한다. 마지막으로 그림 3 (d)와 같이 1/4부화소 보간은 5×5개의 참조화소와 앞서 구한 수평·수직·대각선 1/2부화소를 Bilinear 필터 연산하여 4×4개의 1/4부화소를 생성한다. 따라서 그림 4와 같이 4×4 블록 보간 화소를 구하기 위해서는 1×9 정수화소를 9회에 걸쳐 참조해야 하고, 수평 1/2 부화소 보간 연산을 9번을 수행하며, 수직, 대각선 1/2 부화소 보간 연산을 각 4회, 1/4부화소 보간 연산은 1회 수행해야 한다.

제안하는 움직임보상은 부화소 보간 연산 단위를 4×4 블록화소 단위로 병렬 확장하여 연산처리성능을 높이고 보간 연산 과정에서 사용되는 참조화소들을 공유하여 메모리 검색량을 줄이고자 하였다. 그림 5와 같이 4×4 블록 병렬 보간 연산 연산과정은 수평·수직 1/2부화소 보간, 대각선 1/2부화소 보간, 1/4부화소 보간으로 3단계로 구성된다. 먼저 수평·수직 1/2부화소 보간 연산은 9×9개의 참조화소를 공유한다. 9×9개의 참조화소를 4×9개의 6-Tap FIR 필터 연산하여 4×9개의 수평 1/2부화소를 구하고, 9×9개의 참조화소 중 5×9개의 참조화소를 5×4개의 6-Tap FIR 필터 연산하여 5×4개의 수직 1/2부화소를 구한다. 대각선 1/2부화소 보간에는 4×9개의 수평 1/2부화소를 4×4개의 6-Tap FIR 필터 연산하여 4×4개의 대각선 1/2부화소를 구한다. 1/4부화소 보간은 Separate 1-D FIR 보간 연산과 마찬가지로 5×5개의 참조화소와 앞서 구한 수평·수직·대각선 1/2부화소를 Bilinear 필터 연산하여 4×4개의 1/4부화소를 생성한다. 그림 6는 1/2부화소 및 1/4부화소 보간 연산의 4×4 블록 병렬화된 연산과정을 보인다. 부화소 보간 연산 단위를 4×4 블록단위로 확장시키면 보간 연산을 위한 참조화소들의 공유가 가능하며, 기존 1/2부화소 및 1/4부화소 보간 연산 과정이 반복되던 것을 1회로 줄여 연산처리 성능이 향상시킬 수 있다.

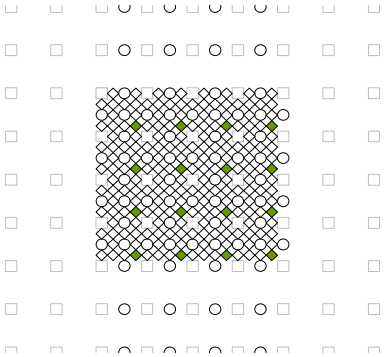
본 연구에서는 보간 연산의 라인단위의 병렬화뿐만 아니라, 블록단위의 병렬처리를 할 수 있는 4×4블록크



(a) 수평・수직 1/2부화소 병렬보간



(b) 대각선 1/2부화소 병렬보간



(c) 1/4부화소 병렬보간

그림 5. 4×4 블록 병렬보간 연산
Fig. 5. 4×4block parallel Interpolation.

기의 병렬 보간 연산인 4×4블록 병렬보간 필터를 제안한다. 실제로 매크로블록 단위에 대해 표 1은 H.264의 가변 블록을 지원할 수 있는 규격의 참조화소의 입력량, 입력횟수 및 보간 연산의 연산기 개수, 연산횟수, 매크로블록당 처리량과 처리성능 비율을 병렬화 적용되지 않은 경우(표준안)과 적용된 경우(Sep. 1-D, 4×4블록)에 대해서 비교 분석하였다. 4×4블록 병렬보간 필터는 모든 부화소 보간 연산을 1회로 처리하여서 매크로 블록에서 단지 1,408의 연산량을 필요로 한다. 이는 표준안의 연산결과의 공유가 없을 때보다 처리성능 비율이

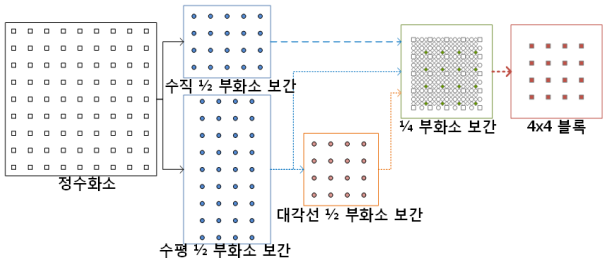


그림 6. 4×4 블록 병렬보간 연산과정
Fig. 6. Process of 4×4 block parallel Interpolation.

표 1. 보간 연산 병렬화에 따른 참조화소 개수 및 보간 연산 횟수 비교
Table 1. Comparison of reference pixel and interpolator for parallel interpolation.

보간 연산	참조화소		보간 연산			처리성능 비율
	입력량 (화소)	입력 (회)	연산기 (개)	연산횟수 (회)	매크로블록당 연산량	(표준안/ Sep.1-D/ 4×4블록)
표준안	6×6	16	4	64	4,096	(1/ 0.6/ 0.34)
Sep. 1-D	9×1	9	17	9	2,448	(0.6/ 1/ 1.74)
4×4블록	9×9	1	88	1	1,408	(2.91/ 1.74/ 1)

2.91이나 개선하였으며, Separated 1-D FIR필터보다 1.74이상 처리성능향상 효과를 보였다.

Ⅲ. 4×4 블록 병렬 보간 움직임보상기 아키텍처

1. 움직임보상기 아키텍처 설계

제안된 4×4 블록 병렬 움직임보상기는 그림 7와 같이 2D 캐쉬 버퍼와 4×4 블록 병렬 보간 필터로 구성된다. 2D 캐쉬 버퍼는 4×4 블록단위 보간 연산에 필요한 참조 화소(9×9 화소)를 공급하는 버퍼이며, 4×4 블록 병렬 보간 필터는 수평·수직·대각선 1/2부화소와 1/4부화소를 생성하는 연산기이다. 2D 캐쉬 버퍼는 움직임 벡터에 따른 중복적인 참조메모리 검색을 최소화하기 위하여 검색영역 간 재사용 기법[11]을 사용한다. 검색영역 간재사용 기법은 이웃한 수평·수직 검색영역 간 중첩되는 부분을 재사용하기 위한 방법으로 외부메모리에 대한 중복검색을 1회로 감소시킬 수 있다. 또한 2D 캐쉬 버퍼는 4×4 블록단위 보간 연산에 사용되는 참조 화소(9×9 화소)를 공급하기 위해 병렬 출력이 가능한 메모리 어레이 구조로 설계되었다. 2D 캐쉬 버퍼와 4×4 블록 병렬 보간 필터는 필요한 9×9개의 참조데이터 1:1로 연결되어 4×4 블록 병렬 움직임보상기를 구성한다. 4×4 블록 병렬 보간 필터는 수평·수직·대각선 1/2부화소와 1/4 부화소를 평면형태로 생성하기 위해 3단 평면 보간 연산 파이프라인 구조를 갖는다. 1단계에서는 수

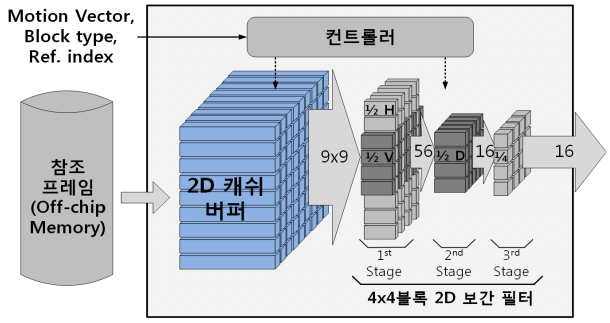


그림 7. 4×4 블록 병렬 움직임보상기 구조
Fig. 7. 4×4 block parallel motion compensation architecture.

평·수직 1/2부화소 보간을 위해 56개의 FIR 필터가, 2단계에서 대각선 1/2부화소 보간을 위해 16개의 FIR 필터가, 3단계에서는 1/4부화소 보간을 위해 16개의 Bilinear 필터가 평면 형태로 할당되어 파이프라인 동작으로 처리성능을 높이고자 하였다.

2. 2D 캐쉬 버퍼

2D 캐쉬 버퍼는 4×4 블록단위 보간 연산에 필요한 참조 화소(9×9 화소)를 공급하는 버퍼이며, 움직임 벡터에 따른 중복적인 참조메모리 검색을 최소화하기 위하여 검색영역 간 재사용 기법을 사용한다. 검색영역 간 재사용 기법은 외부메모리에 대한 검색량을 줄이기 위해 현재의 검색영역과 이웃한 수평·수직의 검색영역과 중첩되는 부분을 미리 저장하여 다음 검색영역에서 재사용하는 방법이다. 그림 8에서 참조 프레임의 크기는 Height와 Width로 표현되며, 검색영역의 한 면의 크기는 M으로, 매크로블록의 한 면의 크기는 N으로 나타내었다. 먼저 수평으로 인접한 검색영역 #1과 #2의 경우 검색영역 #1의 좌측 (M-N)×M를 재사용하고 검색영역 #2의 우측 N×M을 외부메모리에서 검색하게 되면 외부메모리 검색량을 줄일 수 있다. 또한 검색 영역 #5는 수직으로 인접한 검색영역 #4의 M×(M-N)를 재사용하고 부족한 하단의 M×N영역이 요구되지만 이미 검색영역 #3에서 검색영역 #5에 필요한 (M-N)×N이 재사용 가능하여 M×N보다 훨씬 적은 N×N의 외부메모리 검색으로 검색을 줄일수 있다. 따라서 검색영역 간 재사용 기법을 사용하기 위해 2D 캐쉬 버퍼에 “Width×(M+N)” 크기의 참조 데이터를 저장하고, 매크로 블록에 따라 검색영역이 이동할 때마다 N×N만큼의 참조 데이터를 외부메모리로부터 2D 캐쉬 버퍼로 이동하면 외부 메모리의 검색을 최소로 줄일 수 있다.

2D 캐쉬 버퍼는 매크로블록에 따라 검색영역이 이동

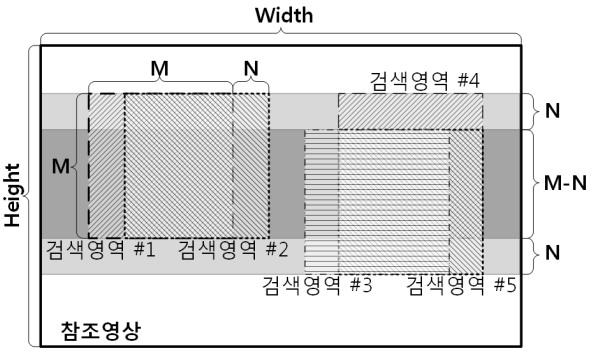
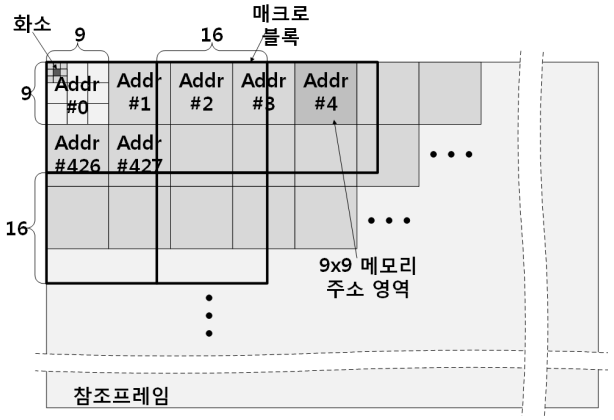
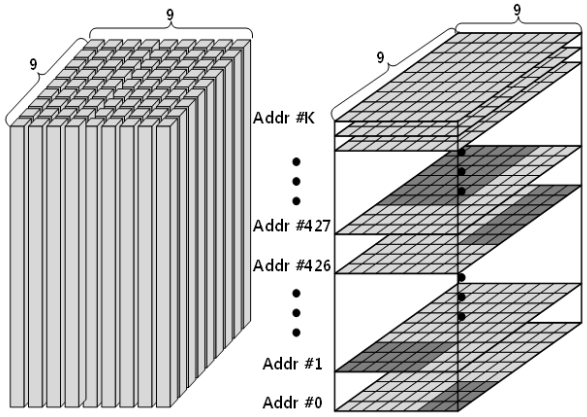


그림 8. 검색영역간 재사용
Fig. 8. Search Area stripe reuse.



(a) 참조프레임에 대한 2D 캐쉬 버퍼 논리적 사상



(b) 참조프레임에 대한 2D 캐쉬 버퍼 물리적 사상

그림 9. 참조프레임에 대한 2D 캐쉬 버퍼 사상
Fig. 9. 2D Cache buffer mapping on reference frame.

할 때마다 매크로블록(16×16) 크기의 데이터를 저장하고, 4×4 블록단위 보간 연산을 위해 9×9 화소 크기의 데이터를 출력한다. 움직임 보상은 매크로블록단위로 수행되지만 가변블록크기를 지원하기 용이하도록 최소블록인 4×4블록 단위로 나누어 처리된다. 4×4블록 단위의 움직임 보상은 9×9개의 참조화소를 2D 캐쉬 버퍼에 16회 요구하기 때문에 1회 저장되는 매크로블록단위가 아닌 그

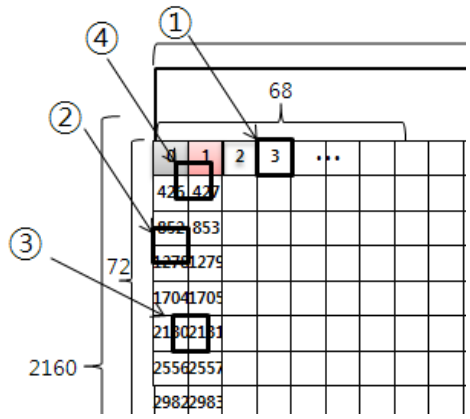
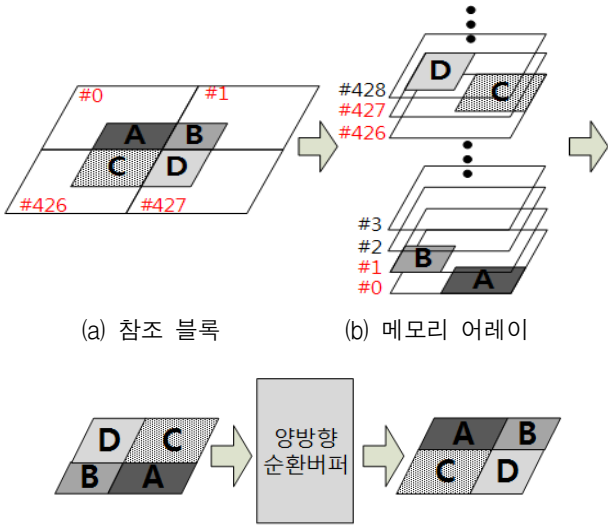


그림 10. 2D 캐쉬 버퍼의 4가지 주소 형태
Fig. 10. 4 type address of 2D Cache buffer.



(a) 참조 블록 (b) 메모리 어레이

(c) 정렬되지 않은 출력 (d) 정렬된 출력

그림 11. 참조블록 재정렬 과정
Fig. 11. Reordering4 of reference block.

림 9 (a)와 같이 참조 프레임을 9×9 정수화소 단위로 분할하고 그림 9 (b)와 같이 9×9개의 메모리 어레이에 구성하여 저장하였다. 이에 따라 보간 연산에 필요한 9×9개의 참조화소를 동시에 검색하기에 효과적인 구조를 갖는다. 그러나 2D 캐쉬 버퍼는 매크로블록 단위로 저장된 외부 메모리의 메모리 사상과 적합하지 않아 최소 4개의 쓰기 주소를 갖는다.

2D 캐쉬 버퍼는 움직임벡터에 따라 보간 연산에 필요한 9×9 정수화소를 4×4 블록 병렬 보간 필터에 공급한다. 그림 10과 같이 움직임벡터에 따라 2D 캐쉬 버퍼 읽기 주소는 4가지 주소형태(①- 81개 모두 동일주소, ②-좌우분할 주소, ③-상하분할 주소, ④-4분할 주소)를 갖는다. 2D 캐쉬 버퍼 읽기 주소가 모두 동일주소일 경우(①)에는 2D 캐쉬 버퍼의 출력되는 화소와

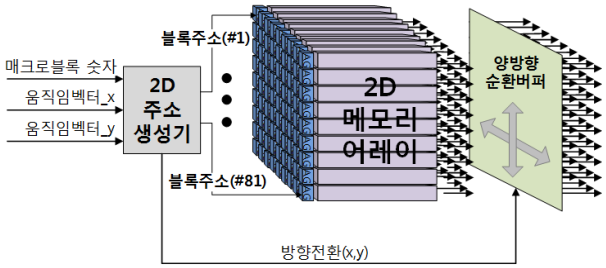


그림 12. 2D 캐쉬 버퍼4의 구조
Fig. 12. 2D Cache architecture and address.

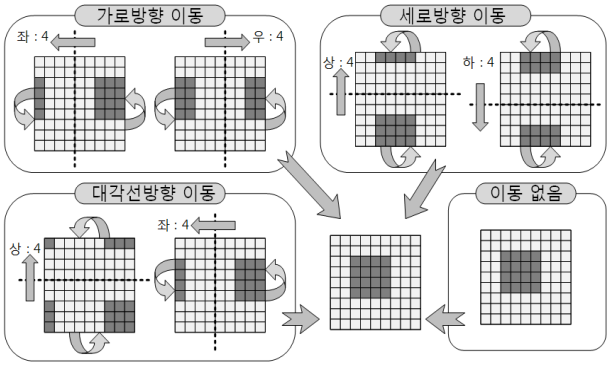


그림 13. 양방향 순환버퍼의 동작
Fig. 13. Bi-directional circular buffer operation.

보간 연산에 입력되는 화소 간 정렬이 맞지만, 2D 캐쉬 버퍼 읽기 주소가 분할주소를 갖는 경우(②, ③, ④)에는 2D 캐쉬 버퍼의 출력되는 화소와 보간 연산에 입력되는 화소 간 정렬이 어긋난다. 그림 11 (a)에서와 같이 4분할 주소를 갖는 경우에는 그림 11 (b)와 같이 9×9 어레이 메모리에 데이터를 요청하고 그림 11 (c)와 같이 정렬된 데이터가 2D 캐쉬 버퍼에 출력된다. 따라서 4분할주소를 갖는 경우 2D 캐쉬 버퍼는 그림 11 (d)와 같이 데이터를 재정렬해야 하는 과정이 필요하며, 그림 12와 같이 2D 캐쉬 버퍼는 움직임벡터에 따라 보간 연산에 필요한 9×9 정수화소를 출력하는 2D 메모리 어레이와 9×9 정수화소를 재정렬하는 양방향 순환버퍼로 구성된다.

움직임벡터에 따라 보간 연산에 필요한 9×9 정수화소를 출력하기 위해 2D 메모리 어레이는 매크로블록 숫자와 움직임벡터(x, y)를 이용하여 2D 메모리 어레이의 읽기 주소를 생성한다. 매크로블록 숫자와 움직임벡터(x, y)에 따라 2D 메모리 어레이의 읽기 주소는 4가지 주소 형태를 가지며 경우에 맞춰 2D 메모리 어레이의 각 메모리에 읽기주소가 공급된다. 양방향 순환 버퍼는 2D 메모리 어레이의 주소가 분할 주소를 갖는 경우 9×9개의 참조화소를 재정렬하여 보간 필터에 전달한다. 그림 13은 양방향 순환버퍼의 동작을 나타내고 있다. 2D 메모리 어레이의 읽기주소가 좌우분할주소를 갖는 경우에는 2D 메모리 어레이의 데이터를 가로방향으로 정렬하고, 상하 분할주소를 갖는 경우에는 2D 메모리 어레이의 데이터를 세로방향으로 정렬하고, 4분할주소를 갖는 경우에는 2D 메모리 어레이의 데이터를 대각선 방향으로 정렬시켜 보간 연산기가 요구하는 형태로 참조화소들을 재정렬한다.

3. 4×4 블록 병렬연산 보간 연산기

4×4 블록 병렬연산 보간 연산기는 2D 캐쉬 버퍼로부터 보간 연산을 위해 필요한 9×9개의 참조화소를 입력받아 4×4 예측 블록을 생성한다. 그림 14는 3단 파이프라인으로 동작하는 4×4 블록 병렬연산 보간 연산기의 구조를 보여준다. 각 단계의 입출력을 살펴보면 평면형태의 부화소가 생성됨을 알 수 있다. 첫 번째 단계에서 9×9 정수화소를 입력받아 4×9개의 수평 1/2부화소와 4×5개의 수직 1/2부화소를 동시에 생성한다. 두 번째 단계에서 첫 번째 단계에서 생성한 4×9개의 수평 1/2 부화소를 입력받아 4×4개의 대각선 1/2부화소를 생성한다. 마지막으로 세 번째 단계에서 앞 서 생성한 1/2화소들과 정수화소들을 입력받아 4×4개의 1/4 화소들을 생성한다. 그리고 마지막으로 4×4개의 정수화소, 수평방향 1/2 화소, 대각선 방향 1/2화소, 수직방향 1/2 화소 및 1/4 화소들 중 하나를 선택하여 4×4의 예측 블록을 생성한다.

각 단계에서의 연산기 구조를 살펴보면 그림 15와 같다. 1단 평면 FIR 필터는 수평·수직 1/2부화소를 동시에 생성할 수 있도록 수평 6 Tap FIR 필터 36개를 4×9 어레이 형태로 구성하고 수직 6 Tap FIR 필터 20개를 5×4어레이 형태로 배치하였다. 1단 평면 FIR 필터는 9×9개의 참조화소를 수평방향 1/2보간 연산하여 4×9개의 수평 방향 1/2화소를 생성하고, 9×9개의 참조화소 중 5×9개의 참조화소를 수직방향 1/2보간 연산으로 5×4개의 수직 방향 1/2화소를 생성한다. 대각선 1/2부화소

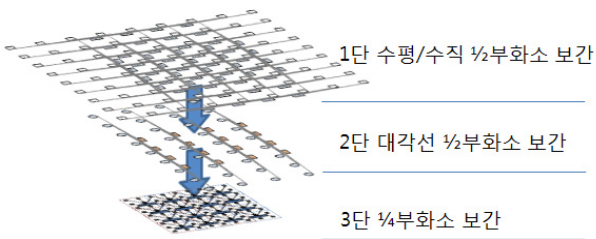
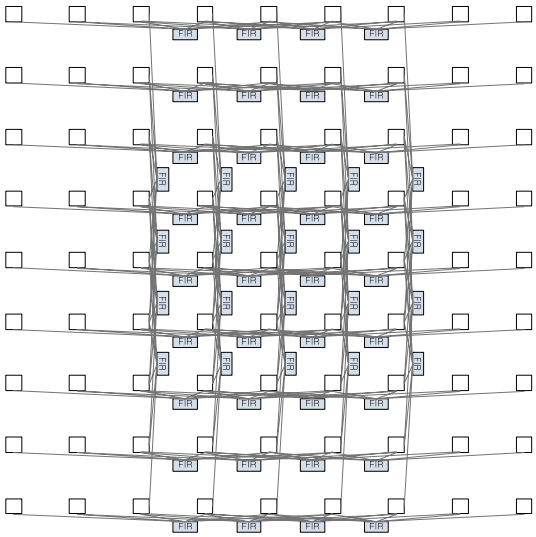
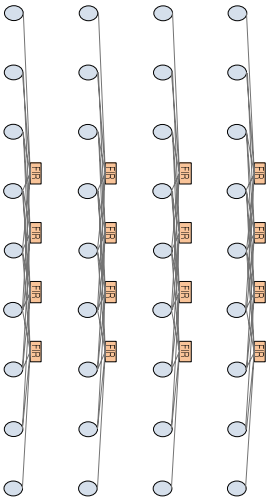


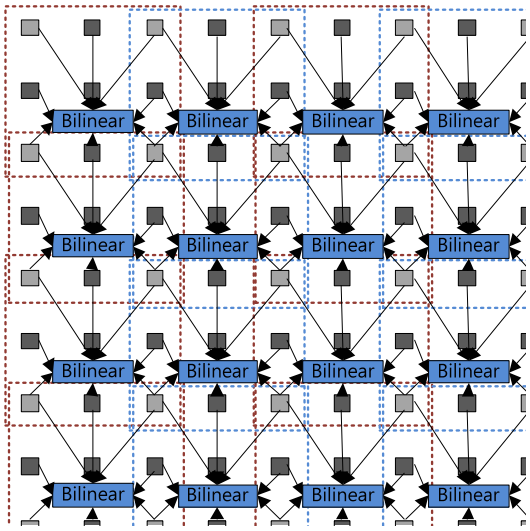
그림 14. 3단 파이프라인구조의 보간 연산기 구조
Fig. 14. Three step pipelined interpolation architecture.



(a) 수평 및 수직 FIR연산



(b) 대각선 FIR연산



(c) 1/4 Bilinear연산

그림 15. 보간 연산 구성
Fig. 15. Configuration of Interpolation.

를 생성하는 2단 평면 FIR 필터는 수직 6 Tap FIR 필터 16개를 4×4어레이로 구성되었다. 1단 평면 FIR 필터로 부터 4×9개의 수평방향 1/2화소를 입력받아 수직방향으로 1/2보간 연산을 수행하여 대각선방향의 1/2화소 4×4개를 생성한다. 마지막 3단 평면 Bilinear 필터는 4×4어레이 구조로 Bilinear 필터 16개를 구성하여 앞 서 생성한 1/2화소들과 정수화소들을 입력받아 1/4화소들을 생성하도록 설계하였다.

Ⅳ. 실험 및 분석

제안된 4×4 블록 병렬 움직임보상기를 0.13um 공정에서 시뮬레이션하여 표 2에 기존 연구와 비교하였다. 먼저 메모리 대역폭을 매크로블록당 검색 사이클과 메모리 크기로 비교해보면 연구[7]는 인접한 서브 블록 간에 입력된 참조화소의 일부를 연산기내에 레지스터를 저장함으로써 다음 서브블록의 데이터 입력 지연시간을 줄였다. 연구[4]는 인접한 매크로블록 2개간의 참조 화소 영역을 통합하는 MB level unification으로 인해 2KBytes 크기의 메모리를 갖는다. 연구[9]는 연구[4]를 확장시켜 인접한 매크로블록 3개를 통합하여 참조영역에 대한 36KBytes의 메모리 크기를 가져 검색 사이클을 105회로 줄인 반면, 제안하는 2D 캐쉬 버퍼는 수평/수직 검색영역 모두를 저장해 234Kbytes의 메모리 크기를 요구하여 연구[9]보다 약 6.5배 더 큰 메모리를 사용함으로써 면적이 증가되었다. 그러나 2D 캐쉬 버퍼는 모든 참조 화소에 대해 한번에 검색이 가능하고 9×9 메모리 어레이 구조를 적용하여 대역폭을 9배로 증가시켜 검색 사이클 수가 15.2%로 감소되므로 짧은 검색 사이클과 높은 대역폭을 제공할 수 있다.

또한 보간 연산기의 병렬성을 FIR 필터의 개수로 비교해보면 기존 Separated 1-D FIR 필터구조[7]는 1-D FIR 필터구조[4]의 임계경로를 줄였으나 움직임벡터에 따라 임계경로가 가변적이여서 두 연구 모두 Full-HD로 처리성능이 제한된다. 연구[9]는 기존 Separated 1-D FIR 필터구조[7]를 사용하면서 병렬화를 2배로 늘려 24개의 FIR 필터로 구성되어 매크로블록당 123사이클이 소요되는 반면, 제안하는 구조는 72개의 FIR 필터를 사용하여 16사이클로 처리성능을 가지며 논문[9]의 비해 7.6배 높은 연산처리성능(Throughput)을 보이고 있다. 또한 연구[9]와의 면적을 게이트 수로 비교해 본 결과, FIR 필터 크기와 관련된 보간 연산기의 크기는 기존 연구 대비 2.6배 커졌으며, 움직임보상기의 크기는 1.4배

표 2. 기존 움직임보상 성능비교
Table 2. Performance comparison for Motion Compensation.

아키텍처		1-D[4]	Separate 1-D[7]	Separate 1-D[9]	4×4 블록 (Proposed)
메모리 접근 기법		MB level unification	Horizontal -switch approach	MB-row level unification	Search area stripe reuse
메모리크기 (KBytes)		2	n/a	36	234
매크로블록당 검색 사이클수		n/a	391.8	105	16
보간 연산기	개수	1	1	2	1
	구성	4 FIR	4 H FIR/ 9 V FIR	4 H FIR/ 8 V FIR	36 H FIR/ 20 V FIR/ 16 D FIR
매크로블록당 클럭 사이클수		304	560	123	16
공정(um)		0.18	0.18	0.13	0.13
면적 (gates)	보간 연산기	13,027	20,686	14,960	77,748
	움직임 보상기	32,000	43,000	117,177	161,435
동작주파수 (Mhz)		100	100	200	150
처리해상도		1920×1088	1920×1088	3840×2160	3840×2160
실시간 복호화 성능		FHD @30fps	FHD @30fps	4K UHD @30fps	4K UHD @72fps

커진 결과이다. 이 같은 결과는 보간 연산기가 움직임보상 내에서의 크기에 영향이 크지 않음을 보이는 결과이다. 마지막으로 움직임보상기의 처리성능을 비교해보면, 기존 연구[9]는 4K UHD 동영상을 30fps로 실시간 처리하기 위해 200MHz라는 높은 주파수를 요구하는 반해, 제안하는 4×4 블록 병렬 보간 움직임보상기는 4K UHD 동영상을 72fps로 실시간 처리하는데 150MHz라는 낮은 주파수를 요구한다. 따라서 제안된 4×4 블록 병렬 움직임보상기는 4K UHD 영상을 실시간 처리함은 물론, 동작주파수 또한 낮아 전력소모를 줄일 수 있다.

V. 결 론

본 연구에서는 H.264 4×4 블록 병렬 움직임보상기를 제안하여 4K UHD 영상을 실시간 처리하는데 주파수를 낮추고 전력소모를 줄이고자 하였다. 연산처리성능을 향상시키고자 보간 연산을 4×4 블록단위로 병렬화하여 매 사이클 마다 4×4 블록 보간 화소를 생성하며 4×4 블록단위 보간 연산구조에서 요구되는 참조 화소를 공급하기 위해 2D 캐쉬 버퍼를 제안하였고, 외부메모리에 대한 검색을 줄이고자 검색영역 간 재사용 기법을 사용하였다. 제안된 4×4 블록 병렬 보간 움직임보상기 구조를 0.13um공정에서 시뮬레이션한 결과, 게이트 수는 161K게이트이며, 동작주파수는 150MHz이다. 이 같은 결과는 기존 연구들에 비해 1.4배의 면적이 증가하였으나, 연산처리성능은 7.6배 향상되어 4K UHD 영상을 72fps로 실시간 처리함을 보였다.

참 고 문 헌

[1] Xue Quan, Liu Jilin, "H.264/AVC Baseline Profile Decoder Optimization on Independent Platform", WiCOM, 2005

[2] RongGang Wang, JinTao Li, Chao Huang, "Motion compensation memory access optimization strategies for H.264/AVC decoder", ICASSP, 2005.

[3] Shih-Hao Wang, Wen-Hsiao Peng, Yuwen He, Guan-Yi Lin, Chen-Yi Lin, Shih-Chien Chang, Chung-Neng Wang, and Tihao Chiang, "A Platform-Based MPEG-4 Advanced Video Coding(AVC) Decoder with Block Level Pipelining", PCM, 2003.

[4] Yu Li, Yanmei Qu, Yun He, "Memory Cache Based Motion compensation Architecture for HDTV H.264/AVC Decoder", Department of Electronic Engineering, Tsinghua University, 2007.

[5] Deng Lei, Gao Wen Hu MingZeng, Ji Zhen Zhou, "An Efficient VLSI Architecture for MC interpolation in AVC Video coding", ESA/VLSI, 2004.

[6] Ronggang Wang, Mo Li, Jintao Li, Yongdong Zhang, "High Throughput and Low Memory Access Sub-pixel Interpolation Architecture for H.264/AVC HDTV Decoder", IEEE Transactions on Consumer Electronics, 2005.

[7] Sheng Zen Wang, "A new motion compensation design for H.264/AVC decoder ", ISCAS, 2005.

[8] Jia-Wei Chen, Chien-Chang Lin, Jiun-In Guo, Jinn-Shyan Wang, "Low Complexity Architecture Design of H.264 Predictive Pixel Compensator for HDTV Application", IEEE ICASSP 2006 14-19 May 2006.

[9] Ping Chao, Youn-Long Lin, "A motion compensation system with a high efficiency reference frame pre-fetch scheme for QFHD H.264/AVC decoding", ISCAS 2008, pp : 256-259

[10] 이찬호, "H.264 복호기를 위한 효율적인 예측 연산기 설계", 대한전자공학회, 전자공학회논문지-SD 46(7), 2009.7, pp : 47-52

[11] Jen-Chieh Tuan, Tian-Sheuan Chang, Chein-Wei

Jen, "On the data reuse and memory bandwidth analysis for full-search block-matching VLSI architecture", IEEE Transactions on Circuits and Systems for Video Technology. Jan 2002 Vol.12 pp : 61-72

저 자 소 개

이 경 호(정회원)
2004년 광운대학교 정보통신 공학과 학사 졸업.
2006년 광운대학교 컴퓨터공학과 석사 졸업.
2006년~현재 광운대학교 컴퓨터 공학과 박사과정
<주관심분야 : 영상신호처리, SoC설계, VLSI, Embedded System>

공 진 흥(평생회원)
1980년 서울대학교 전자공학과 학사 졸업.
1982년 한국과학기술원 전기 및 전자공학과 석사 졸업.
1989년 텍사스주립대학교 컴퓨터공학과 박사 졸업.
<주관심분야 : 영상신호처리, SoC설계, VLSI, Embedded System>