논문 2009-46SD-7-7

H.264 복호기를 위한 효율적인 예측 연산기 설계

(Design of Prediction Unit for H.264 decoder)

이 찬 호*

(Chanho Lee)

요 약

H.264 영상 압축 표준은 높은 압축률과 화질로 널리 이용되고 있다. 이러한 H.264 복호기에서 움직임 보상기는 가장 연산 시간이 오래 걸리고 복잡한 유닛이다. 이러한 움직임 보상기의 성능은 보간 연산기와 참조 픽셀을 외부에서 읽어 오는 동작의 효율성에 의해 결정된다. 따라서 고성능 보간 연산기를 설계하고 참조 메모리와 데이터의 관리를 통해 데이터 재활용을 늘려 외부 메모리 접근 횟수를 줄이는 것이 필요하다. 본 논문에서는 2 차원 회전 레지스터 파일과 움직임 벡터 예측기, 그리고 저 복잡도 고성능의 보간 연산기를 이용한 효율적인 움직임 보상기 구조를 제안한다. 2 차원 회전 레지스터는 참조 메모리에서 읽어 온 픽셀 데이터를 보관하면서 보간 연산기에 필요한 픽셀 데이터를 신속하게 공급하고 재활용될 데이터를 효과적으로 처리할 수 있는 기능을 가지고 있다. 제안된 구조에 따라 움직임 보상기를 설계하고 인트라 예측기와 통합하여 예측 연산기를 구현하여 동작과 성능을 검증하였다.

Abstract

H.264 video coding standard is widely used due to the high compression rate and quality. The motion compensation is the most time-consuming and complex unit in the H.264 decoder. The performance of the motion compensation is determined by the calculation of pixel interpolation and management of the reference pixels. The reference pixels read from external memory using efficient memory management for data reuse is necessary along with the high performance interpolators. We propose the architecture of a motion compensation unit for H.264 decoders. It is composed of 2-dimensional circular register files, a motion vector predictor and high performance interpolators with low complexity. The 2-dimensional circular register files reuse reference pixel data as much as possible, and feed reference pixel data to interpolators without any latency and complex logic circuits. We design a motion compensation unit and a intra-prediction unit, and integrate them into a prediction unit and verify the operation and the performance..

Keywords: H.264 Decoder; Motion Compensation; Interpolator; circular register files

Ⅰ. 서 론

H.264는 ISO/IEC와 ITU-T의 JVT(Joint Video Team)에 의해 개발된 동영상 압축 표준으로 MPEG-4 part 10 AVC라고도 불린다[1]. H.264의 압축률은 MPEG-4 ASP(Advanced Simple Profile)보다 25-40% 정도 더 좋다[2]. 그러나 연산 알고리즘이 복잡하여 복호 연산의 경우 MPEG-4에 비해 연산 복잡도가 2배 정도 증가하였다. H.264 복호 소프트웨어를 이용한 실행 프로파일 분석 결과에 의하면 움직임 보상기(MC: Motion Compensation)는 전체 복호 시간 중에서 최대 55%까지 차지한다. 따라서 움직임 보상기의 구조를 최적화하는 것이 복호기 성능에 매우 큰 영향을 미친다는 것을 알 수 있다[3~4].

H.264 복호기의 픽셀 예측 유닛은 동일 프레임 내에서 이웃의 픽셀 데이터로부터 값을 예측하는 인트라 예측기와 다른 프레임의 픽셀 데이터로부터 값을 예측하는 움직임 보상기로 구성된다. 인트라 예측기는 동일 시간에 동일 물체에서 인근 픽셀값이 유사한 것을 이용한 것이고 움직임 보상기는 다른 시간대의 동일 물체에

* 정회원, 숭실대학교 정보통신전자공학부
(School of Electronic Engr., Soongsil University)
※ 본 논문은 ETRI SoC산업진흥센터에서 수행한 ITSoC 핵심설계인력양성사업의 지원을 받았습니다.
접수일자: 2009년4월25일, 수정완료일: 2009년7월1일

대해 움직임을 예측하여 해당 픽셀값을 계산한다. 모든 픽셀 값은 인트라 예측기 또는 움직임 보상기에 의해 계산되므로 P 슬라이스의 경우 I 매크로블록과 P 매크로블록이 연속하여 나타나는 경우 적절한 제어를 통해 두 예측기가 동시에 동작하도록 하여 복호기의 성능을 높일 수 있다.

일반적으로 I 슬라이스에 비해 P 슬라이스가 훨씬 많고 P 슬라이스 내에서도 P 매크로블록이 I 매크로블록보다 훨씬 많으므로 대부분의 예측 연산은 움직임 보상기에 의해 이루어지고 움직임 보상 연산은 인트라 예측기보다 훨씬 복잡하여 복호 성능은 움직임 보상기의 성능에 의해 결정된다. 움직임 보상기는 움직임 벡터 예측기(MVP: Motion Vector Predictor), 주소 생성기와 참조 픽셀 수집기, 그리고 보간 연산기(Interpolator)로 구성된다. 주소 생성기는 움직임 벡터를 이용하여 외부 메모리의 참조 픽셀 데이터의 주소를 계산하여 데이터를 읽어올 수 있게 한다. 움직임 보상기의 성능은 외부 메모리에 접근하여 데이터를 읽어오는 사이클 수에 영향을 받으므로 이를 줄이는 것이 중요하다[5~6]. 휘도 데이터의 경우 보간 연산기는 4x4 서브 블록을 예측하기 위해 9x9 참조 픽셀이 필요하고 움직임 벡터는 1/4 픽셀 위치까지 표현한다. 1/2 픽셀이나 1/4 픽셀 위치의 데이터는 정수 위치의 픽셀 데이터를 이용하여 보간 연산을 통해 계산한다. 즉, 1/2 픽셀 위치의 데이터는 가로나 세로의 6 개의 정수 위치 픽셀 데이터를 6 탭 FIR 필터를 이용하여 계산하고 1/4 픽셀 위치 데이터는 1/2 픽셀값과 정수 픽셀값을 이용하여 반올림한 평균값인 클립(Clip) 연산을 통해 얻는다. 색차 데이터는 네 개의 정수 픽셀값을 이용하여 1/8 픽셀값을 이중선형(bilinear) 보간 필터를 통해 계산한다. 보간 연산은 픽셀의 위치에 따라 최대 7 번의 6 탭 필터 연산과 한번의 클립 연산이 필요하므로 데이터를 공급하는 유닛을 포함한 보간 연산기는 움직임 보상기의 성능을 결정한다. 따라서 움직임 보상기의 성능을 개선하기 위해서는 메모리 접근 사이클 수와 보간 연산기의 연산 사이클을 줄이는 것이 중요하다.

본 논문에서는 H.264 복호기를 위한 효율적인 움직임 보상기 구조를 제안한다. 그 구조는 움직임 벡터 예측기와 주소 생성기, 그리고 2 차원 회전 레지스터 파일과 보간 연산기로 구성되어 있다. 보간 연산기는 이중 채널 파이프라인 구조를 가지고 있고 덧셈기와 쉬프터만으로 정밀도 저하 없이 정수, 1/2, 1/4 픽셀값에 대한 보간 연산을 수행하며, 휘도와 색차 데이터에 대해 별도의 연산기가 존재한다[7]. 2 차원 회전 레지스터 파일은 보간 연산기가 효과적으로 연산을 하도록 정수 픽셀 데이터를 공급하는 역할을 수행한다. 그 구조는 휘도 데이터를 위한 13x9와 9x9 레지스터 파일과 색차 데이터를 위한 두 개의 5x3 레지스터 파일로 구성되고 데이터를 재활용하여 참조 메모리 접근을 최소화하고 지연없이 보간 연산기에 데이터를 공급하기 위해 데이터의 회전(rotation)이 가능하다. 제안된 구조에 따라 움직임 보상기와 인트라 예측기를 설계하고 통합하여 예측 연산기를 구현하였다. 구현된 연산기는 FPGA를 이용하여 동작을 검증하였다.

II. 움직임 보상기의 참조 픽셀

H.264 영상 압축 표준에서 휘도 데이터의 기본 매크로블록의 크기는 16x16 픽셀이지만 인트라 예측기나 움직임 보상기는 4x4 픽셀의 서브 블록 단위로 연산이 가능하다. 또한 움직임 보상기에서 움직임 벡터는 정수의 1/4 위치까지 표현이 가능하여 참조 프레임의 정수 픽셀 데이터에 대해 1/2 및 1/4 픽셀을 계산할 수 있어야 한다. 그림 1에서 K와 L을 제외한 G~Y의 4x4 서브 블록에 대해 움직임 보상 연산을 하는 경우, 첫 번째 픽셀인 G에 대해 움직임 보상기에서 계산해야하는 픽셀은 움직임 벡터의 소수점 아래 값에 따라 m을 제외한 a~r까지의 15 개이다. 15 개의 픽셀은 그 위치에 따라 크게 5 가지의 방법으로 계산된다. 계산해야할 픽셀의 위치 즉, 움직임 벡터의 값에 따라 보간 연산기의 동작

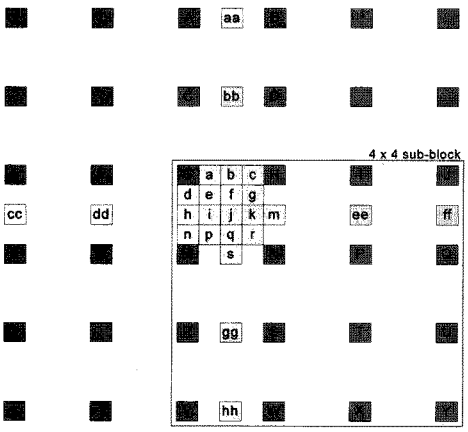


그림 1. 정수, 1/2, 1/4 위치의 휘도 픽셀 데이터
Fig. 1. Integer-, half-, and quarter-pixels of luminance.

이 달라지고 공급해야할 픽셀값의 수와 위치도 다르다. G의 경우 그림 1에 나타난 모든 픽셀이 필요하다. Y의 경우 아래와 오른쪽로 3 줄씩 픽셀값이 더 필요하므로 결국 4x4 서브 블록에 대해 9x9 의 참조 픽셀 데이터가 필요하다. 색차 데이터의 경우 서브 블록의 크기는 2x2 이고 4 개의 정수 픽셀값을 이용하여 1/8 픽셀값을 계산하므로 3x3 크기의 참조 픽셀 데이터만 있으면 하나의 서브 블록에 대한 보간 연산이 가능하다.

앞서 언급한 바와 같이 움직임 보상기의 보간 연산기는 이미 복호된 참조 프레임의 픽셀 데이터가 필요한데 이들은 보통 외부 메모리에 저장된다. 보간 연산기는 다양한 위치에 있는 6 개의 픽셀 데이터가 한 번에 필요하므로 참조 데이터에 대한 메모리 관리는 매우 중요하다. 움직임 보상기의 성능은 보간 연산기 자체의 성능과 함께 참조 데이터 픽셀을 공급하는 효율성에 의해서도 크게 영향을 받는다. 참조 데이터는 일반적으로 외부의 DRAM에 저장되는데, DRAM에 접근하여 참조 데이터를 읽어 오기 위해서는 많은 사이클이 소모되므로 미리 데이터를 읽어 오지 않으면 연산기는 데이터를 기다리는 시간이 길어져 움직임 보상기의 성능은 급격히 감소한다. 따라서 외부 메모리에 접근을 최소화하고 데이터를 미리 읽어 와서 보간 연산기가 준비가 되었을 때 바로 데이터를 공급해주는 것이 중요하다.

III. 움직임 보상기의 구조

제안하는 움직임 보상기는 움직임 벡터 예측기와 주소 생성기, 2 차원 회전 레지스터 파일과 보간 연산기로 구성된다. 그림 2에 나타난 바와 같이 움직임 벡터 예측기는 엔트로피 복호기에서 움직임 벡터 차이값을 받아 움직임 벡터를 계산하여 주소 생성기로 보내 참조 메모리에서 해당 참조 픽셀 데이터를 읽어 오기 위한 주소값을 생성하여 데이터를 요청한다. 참조 픽셀값이 공급되면 2 차원 회전 레지스터 파일에 저장되어 보간 연산기에 데이터를 공급한다. 한편, 움직임 벡터 예측기는 2 차원 회전 레지스터 파일에도 움직임 벡터를 공급하여 재사용할 데이터를 구분하여 다음 서브 블록에서 사용할 수 있도록 내부에서 처리한다.

휘도와 색차 데이터를 위한 보간 연산기는 고속의 연산을 위해 이중 채널 파이프라인 구조를 갖고 있으며 두 보간 연산기가 독립적으로 동작하여 데이터 처리 속도가 빠르다. 색차 보간 연산이 휘도 보간 연산에 비해 상

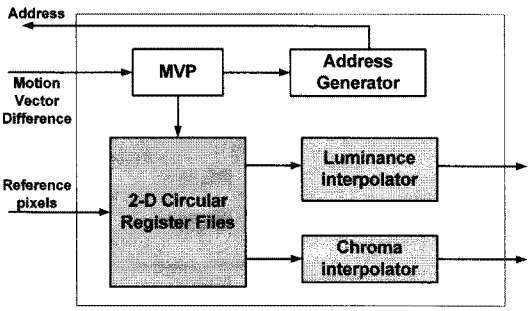


그림 2. 움직임 보상기의 구조
Fig. 2. The architecture of motion compensation unit.

대적으로 빨리 끝나므로 참조 픽셀 데이터를 읽어 올 때 참조 메모리 접근 충돌이 발생하면 휘도 연산이 우선적으로 이루어진다. 휘도 보간 연산은 픽셀의 위치에 따라 한 사이클에 하나 또는 2 개의 픽셀이나 11 사이클에 8 개의 픽셀을 처리할 수 있다[7].

앞에서 언급한 바와 같이 보간 연산기에 픽셀 데이터를 빠르게 공급하기 위해서 움직임 보상기에는 버퍼가 사용되는데 4x4 서브 블록을 위해서는 9x9 크기의 픽셀 데이터가 필요하다. 따라서 하나의 매크로블록에 대해서는 1,296(=9x9x16)개의 픽셀을 외부의 참조 메모리에서 읽어 와야 한다. 데이터 버스가 32 bit인 경우 9x9 픽셀 데이터 읽기는 27 개의 트랜스액션(transaction)에 해당하는데 보간 연산 전에 모두 완료되어야 한다. 버퍼는 이렇게 미리 읽어온 데이터를 저장한다. 데이터를 공급하는데 있어서 또 하나의 문제는 보간 연산기에 다양한 위치에 있는 6 개의 픽셀 데이터를 동시에 공급하는 일이다. 예를 들어 그림 1의 j를 계산하기 위해서는 정수 픽셀값들을 수평 방향으로 한줄 씩 6 번 공급하여 aa, bb, b, s, gg, hh를 구하고 이들이 다시 연산기에 입력되어야 한다. 이러한 과정이 빠르게 이루어지려면 버퍼는 매우 복잡한 구조를 갖게 된다.

제안하는 움직임 보상기 구조에서는 휘도 연산의 경우 4x4 서브 블록에 대해 9x9 2 차원 회전 레지스터 파일이 이용된다. 그림 3에서 A와 B의 4x4 서브 블록에 대한 연산을 위해 두 개의 9x9 참조 픽셀이 필요하다. 그러나 A와 B는 연속해 있으므로 움직임 벡터가 같으면 빗금친 영역에 해당하는 5x9 참조 픽셀 데이터를 공유한다. 매크로블록내의 이웃한 서브 블록은 동일한 움직임 벡터를 가질 가능성이 매우 높은데, 그러한 경우 13x9 레지스터 파일을 이용하면 A와 B처럼 이웃해 있는 두 서브 블록의 참조 픽셀 데이터를 한 번에 가져와

저장할 수 있다. 9x9 픽셀을 2 번 읽어 오는 경우 읽기 동작을 54 회 수행해야 하나 13x9 픽셀을 읽어 오는 동작은 36 회면 충분하다. 그림 3에서 C와 D 서브 블록도 움직임 벡터가 같다면 점선으로 표시된 13x4 영역의 픽셀만 추가로 읽어 오면 된다. 결과적으로 A~D 서브 블록의 움직임 벡터가 같은 경우 픽셀 데이터를 재활용하면 108 회의 읽기 동작이 52 회로 줄어든다. E와 F의 움직임 벡터도 같다면 점선안의 데이터를 읽어 오는 동안 빗금친 영역의 오른쪽 데이터 중에서 5x4 블록을 임시 저장소에 저장한다. 그러면 A~D에 대한 보간 연산이 끝나고 E와 F의 보간 연산을 위해서는 8x9 참조 픽셀 블록만 읽어 오고 나머지 데이터는 빗금친 영역의 오른쪽 데이터를 재활용할 수 있다. 다시 G와 H의 움직임 벡터가 같다면 점선 영역의 5x4 픽셀이 임시 저장소에 저장되고 8x4 픽셀 블록을 읽어 온다. 따라서 A~H의 서브 블록이 동일한 움직임 벡터를 가지면 2 차원 회전 레지스터 파일을 이용하여 216 회의 읽기 동작을 70 회로 줄일 수 있다. 16x16 매크로블록에 대해서는 9x9 회전 레지스터 파일이 하나 더 있다면 432 회의 읽기 동작을 118 회로 줄일 수 있다.

제안한 데이터 재활용 방식을 적용하였을 때 MB 타입에 따라 읽기 동작의 횟수가 얼마나 감소하는 지를 보여주는 자료가 표 1의 왼쪽 두 열에 나타나 있다. 움직임 벡터는 MB 타입에 따라 블록별로 정해지므로 4x4 보다 큰 블록의 내부 4x4 서브 블록은 모두 동일한 움직임 벡터를 갖는다. 표 1에 나타난 값들은 MB 타입에 따른 각 블록의 이웃한 블록은 서로 다른 움직임 벡터를 갖는다고 가정하고 계산한 읽기 동작의 비율이다.

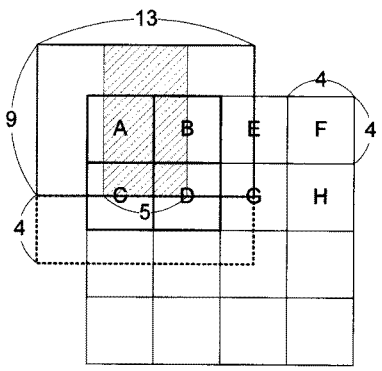


그림 3. 휘도 보간 연산에 필요한 참조 픽셀 데이터 영역

Fig. 3. Reference pixel data ranges for interpolations of luminance pixels.

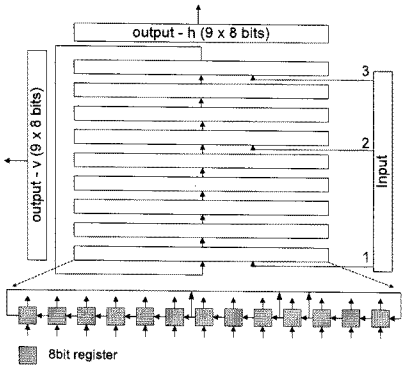
표 1. 제안한 데이터 재활용 방식에서 MB 타입과 테스트 영상에 따른 메모리 접근 비율

Table 1. The reduction rate of memory access with the proposed data reuse scheme according to the MB types and test video clips

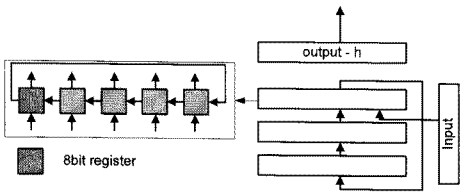
MB type	Rate of Read Transactions	Test Video	Rate of Read Transactions
16x16	0.273	carphone	0.435
16x8	0.324	football	0.410
8x16	0.481	bus	0.398
8x8	0.481	foreman	0.332
8x4	0.667	akiyo	0.415
4x8	0.667	average	0.398
4x4	1.000		

4x4 타입에서 모든 블록이 서로 다른 움직임 벡터를 갖는다고 가정했을 때 읽기 동작의 수를 1이라고 하면 블록 크기가 커질수록 읽기 동작 횟수는 감소하여 16x16 블록은 0.273이 되는데 이는 약 73%의 데이터를 재활용하여 그만큼 메모리 접근 횟수를 줄어든다는 의미가 된다. 표 1의 오른쪽 두 열은 다양한 테스트 영상에 대해 MB 타입의 종류와 수를 분석하여 제안하는 방식을 적용하였을 때 읽기 동작이 얼마나 감소하는지를 보여주고 있다. 자료를 분석해 보면 데이터 재활용율이 54%에서 67%인 것을 알 수 있다. 이 값은 이웃한 블록이 모두 서로 다른 움직임 벡터를 갖는 것을 가정한 것이므로 실제로는 이웃한 블록이 같은 움직임 벡터를 가질 가능성이 크다는 것을 고려하면 실제 재활용율은 80% 이상 될 것으로 추정된다.

앞에서 언급한 데이터 재활용을 하기 위해서는 이를 지원하는 버퍼가 원활하게 동작해야 한다. 이를 위해 2 차원 회전 레지스터 파일의 구조를 제안한다. 구조를 살펴보면 그림 4에 나타난 것처럼 13x9 또는 5x3 배열의 8 비트 레지스터가 수평과 수직 방향으로 모두 데이터를 회전시킬 수 있도록 연결되어 있고 최종 출력값을 저장하는 출력 레지스터가 있다. 그림 2의 보간 연산기는 이중 채널 파이프라인 구조로 되어 있어 한 번에 두 세트의 입력을 받아 한 번에 처리가 가능하다[7]. 보간 연산기의 성능을 최대로 발휘하려면 레지스터 파일이 동시에 두 세트의 입력을 공급할 수 있어야 하므로 제안한 구조에서는 수평 및 수직 출력 레지스터를 통해서 이 기능을 수행한다. 예를 들어 그림 1의 E, F, G, H, I, J를 수평 레지스터(output-h)를 통해서, aa, bb, b, s, gg, hh를 수직 레지스터(output-v)를 통해서 복잡한 재정렬 절차 없이 동시에 공급할 수 있다. 출력 레지스터



(a) 휘도 픽셀 연산을 위한 레지스터 파일



(b) 색차 픽셀 연산을 위한 레지스터 파일

그림 4. 2 차원 회전 레지스터 파일의 구조도
Fig. 4. Proposed 2-dimensional circular register files.

는 9 개의 데이터를 가지고 있으므로 한번 입력을 받으면 보간 연산기에 데이터를 4 번 공급할 수 있다. 이 사이에 내부의 회전 레지스터 파일은 수직 또는 수평 방향으로 데이터를 이동하여 다음에 필요한 데이터를 출력 레지스터로 보낼 준비를 한다. 참조 메모리에서 새로 가져온 데이터는 동작 모드에 따라 적절한 위치에서 입력되어 저장된다. 예를 들어 그림 3에서 B 블록의 움직임 벡터가 A 블록과 다르다면 A 블록에 대한 보간 연산을 하는 동안 B 블록을 위한 참조 픽셀을 읽어와 저장한다. 만일 외부 메모리 접근 시간이 너무 오래 소모되어 다음 블록에 필요한 참조 데이터를 정해진 시간 내에 미리 읽어오기가 어려운 경우에는 휘도 픽셀의 경우 9x9 2 차원 회전 레지스터 파일을 하나 더 추가하면 이중 버퍼링이 가능해져 성능을 개선할 수 있다. 색차 픽셀을 위한 2 차원 회전 레지스터 파일은 데이터 수를 제외하고는 휘도 픽셀용과 큰 차이가 없다.

IV. 설계 및 검증

제안된 구조에 따라 움직임 보상기와 인트라 예측기를 Verilog-HDL을 이용하여 설계하였다. 움직임 보상기는 그림 2의 구조와 같고 인트라 예측기는 기존의 스

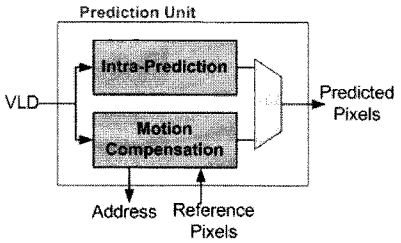


그림 5. H.264 복호기용 예측 연산기 구조
Fig. 5. Proposed prediction unit for H.264 decoders.

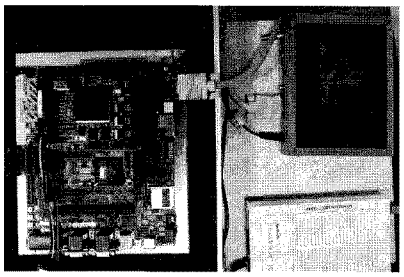


그림 6. FPGA 검증결과
Fig. 6. Result of verification using an FPGA.

표 2. 기존 결과와의 비교 (0.18um CMOS 공정, 초당 30 프레임의 HD 영상처리, MVP 제외)
Table 2. Comparison of synthesized results with other works (0.18um CMOS technology, HD image size at 30fps without MVP)

	[2]	[3]	[9]	본 논문
게이트 수	46,646	64,882	43,030	38,820
동작주파수[MHz]	125	250*	125	115 (250*)

* 최대 동작 주파수

케일러블 구조를 이용하였다[8]. 통합된 예측 연산기의 구조가 그림 5에 나타나 있다. 예측 연산기는 0.18um CMOS 표준 셀 라이브러리와 IDEC에서 지원받은 Synopsys Design Compiler를 이용하여 합성하였고 합성결과 115 MHz에서 HD (1920x1080) 영상을 초당 30 프레임 처리 가능함을 확인하였다. 합성 면적은 인트라 예측기가 20,400 게이트, 움직임 보상기가 움직임 벡터 예측기를 포함하여 52,640 게이트, 움직임 벡터 예측기를 제외하고 38,820 게이트를 차지하였다. 최대 동작 주파수는 250MHz로 UHD(2560x1600) 영상을 초당 30 프레임 처리할 수 있다. 면적을 줄일 필요가 있을 때 보간 연산기와 레지스터 파일을 하나만 사용하여 설계할 경우 202 MHz에서 HD 영상을 초당 30 프레임을 처리할 수 있고, 이 때 게이트 수는 25,532이다. 설계된 예측 연산기는 FPGA에 구현하여 동작을 검증하였고 그 결과 이미지가 그림 6에 나타나 있다.

표 2는 움직임 보상기를 기존의 결과와 비교한 표이다. 모든 결과는 0.18um CMOS 공정을 이용하여 합성하였고 동작 주파수는 [3]의 결과를 제외하고는 HD 영상을 처리하는데 필요한 값이다. 면적은 움직임 벡터 예측기를 제외한 게이트 수이다. 비교 결과 본 논문에서 제안하는 구조가 가장 적은 면적을 차지하면서 요구 동작 주파수가 낮아 참조 픽셀 데이터를 효과적으로 처리하고 있음을 알 수 있다.

V. 결 론

H.264 복호기에서 가장 연산이 복잡하고 시간이 많이 소요되는 움직임 보상기를 위한 효율적인 구조를 제안하였다. 움직임 보상기 내부의 고성능 보간 연산기는 이중 채널 파이프라인 구조를 가지고 있고 이를 지원하는 2 차원 회전 레지스터 파일은 수직 및 수평 방향으로 데이터를 자유롭게 회전시켜 보간 연산기에 데이터를 적절하게 공급한다. 특히 참조 픽셀 데이터의 재활용률을 높여 움직임 보상에서 성능 향상의 가장 큰 장애중 하나인 외부 메모리 접근 횟수를 줄임으로써 연산 사이클 단축과 저전력 소모의 효과를 얻을 수 있다. 시뮬레이션 결과 다양한 테스트 영상에 대해 최소 60% 정도의 재활용률을 보였고 실제 영상에서는 80% 정도까지 재활용이 가능할 것으로 기대된다. 제안된 구조에 따라 움직임 보상기를 설계하고 인트라 예측기와 통합하여 예측 연산기를 구현하고 FPGA에서 동작을 검증하였다. 합성 결과 예측 연산기는 115 MHz에서 1080p HD 영상 처리가 가능하고 최대 250 MHz에서 동작하여 1600p UHD 영상까지 처리가 가능하다. 움직임 보상기는 합성후 면적이 38,816 게이트로 기존의 결과에 비해 11 - 67% 정도 적은 게이트 수로 구현 가능함을 확인하였다.

참 고 문 헌

[1] Joint Video Team, Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification. ITU-T Rec. H.264 and ISO/IEC 14496-10 AVC, May 2003.

[2] Chuan-Yung Tsai, Tung-Chien Chen, To-Wei Chen, Liang-Gee Chen, "Bandwidth Optimized Motion Compensation Hardware Design for H.264/AVC HDTV Decoder" Proceedings of 48th Midwest Symposium on Circuits and Systems, Vol. 2, pp. 1199-1202, Aug. 7-10, 2005.

[3] Mythri Alle et al, "High performance VLSI implementation for H.264 Inter/Intra prediction", Proceedings of IEEE International Conference on Comsumer Electonics, pp.1-2, Jan. 2007.

[4] Wen-Nung Lie, Han-Ching Yeh, Lin, T.C.-I., Chien-Fa Chen, "Hardware-efficient computing architecture for motion compensation interpolation in H.264 video coding", Proceedings of IEEE International Symposium on Circuits and Systems, Vol. 3, pp. 2136-2139, May 23-26, 2005.

[5] 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, Vol. 51, pp. 1006-1013, Aug. 2005.

[6] Sheng-Zen Wang, Ting-An Lin, Tsu-Ming Liu, Chen-Yi Lee, "A new motion compensation design for H.264/AVC decoder" IEEE International Symposium on Circuits and Systems, Vol. 5, pp. 4558-4561, May 23-26, 2005.

[7] Yonghoon Yu, Chanho Lee, "Design of Low-Complexity Interpolator for Motion Compensation in H.264 decoder" Proceedings of International Conference on Circuits/Systems Computers and Communications, pp.225-228, Jul. 2008.

[8] 이찬호, "H.264 복호기를 위한 스케일러블 인트라 예측기 구조 설계," 전자공학회 논문지, 제45권 SD편 제11호, pp.77-81, 2008.11

[9] Jaemoon Kim, Woong Hwangbo, Chong-Min Kyung, "A Novel Bus Interface and Motion Compensation Architecture for H.264/AVC with Reduced Memory Access," 16회 한국반도체학술대회, 대전, 2009.2.20

저 자 소 개

이 찬 호(정회원)
대한전자공학회 논문지
제43권 SD편 제9호 참조
현재 숭실대학교 정보통신전자공학부 교수