H.264/AVC 디코더를 위한 효율적인 인터 예측 하드웨어 구조 설계

김선철*.류광기**

한밭대학교 정보통신전문대학원 정보통신공학과

An Efficient Inter-Prediction Hardware Design for the H.264/AVC Decoder

Xianzhe Jin*·Kwangki Ryoo**

Graduate School of Information and Communication, Hanbat National University

E-mail : suntreekim@gmail.com, kkryoo@hanbat.ac.kr

요 약

본 논문에서는 H.264/AVC 베이스라인 프로파일 가운데서 병목현상을 일으키는 주요한 부분인 인터 예측의 효율적인 하드웨어 구조에 관한 설계에 대해 기술한다. H.264/AVC 디코더는 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4 등 다양한 블록 모드를 지원하는데 레퍼런스 소프트웨어(JM)에서는 중복 픽셀에 대해 제거 하지 않고 항상 4x4 블록에 대한 9x9 참조 블록을 패취하게 된다. 기존에 이미 설계된 디자인에서는 이러한 문제를 해결하기 위하여 8x8 블록 모드와 4x4 블록 모드를 고려하여 설계하였다. 블록 모드가 8x8 사이즈보다 크거나 같을 경우 여러 개의 8x8 블록으로 나뉘어서 그에 대한 13x13 레퍼런스 블록을 패취하고 8x8 블록 보다 작을 경우 여러 개의 4x4 블록으로 나뉘어서 그에 대한 9x9 레퍼런스 블록을 패취하는 방법을 사용하여 중복픽셀을 제거 하여 패취 사이클을 줄였다. 본 논문에서는 더 큰 성능 향상을 위하여 8x8과 4x4 블록 모드뿐만이 아닌 다양한 블록 모드에 대한 레퍼런스 블록 패취를 진행하여 더 많은 중복픽셀을 제거 하였고 메모리 패취 사이클을 줄여 최대 18.6%의 참조 블록 패취 사이클 감소를 가져 왔다.

ABSTRACT

Inter-Prediction is always the main bottleneck in H.264/AVC Baseline Profile. This paper describes an efficient Inter-Prediction hardware architecture design. H.264/AVC decoder supports various block types such as 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4 block types. Reference Software(JM) only considers the 4x4 block type when the reference block is being fetched. This causes duplicated pixels which needs extra fetch cycles. In order to eliminate some of the duplicated pixels, the 8x8 and 4x4 block types were considered in the previous design. If the block size is larger than or equal to the 8x8 block type, it will be separated into several 8x8 block types and if the block size is smaller than the 8x8 block type it will be separated into several 4x4 blocks. For further reduction of the fetch cycles, the various block types are considered in this paper. As a result, the maximum cycle reduction percentage is 18.6% comparing with the previous design.

Ⅰ. 서 론

H.264/AVC 비디오 압축 표준[1]은 Joint Video Team(JVT)에 의해 개발되었고 현존하는 압축률이 가장 우수한 비디오 압축 표준으로 기존의 압축 표준과 비교하여 MPEG-2의 약 2배, MPEG-4 ASP(Advanced Simple Profile)의 약 1.5배의 압축 성능의 향상을 보인다[2]. 그 이유는 가변 블록 크기의 움직임 예측 및 보상, 정수 변환 및 양자화, 1/4 화소단위의 화면 간 예측, 다중 참조 프레임 기반의 움직임 예측 및 보상, 방향성을 띈 화면 내 예측 부호화, 디블로킹 필터,

본 연구는 IDEC의 지원 및 지식경제부 출연금으로 ETRI, 시스템반도체산업진흥센터에서 수행한 IT SoC 핵심설계인력양성사업의 연구결과임.

엔트로피 부호화 등의 다양한 기술들이 향상되었기 때문인데, 그 가운데서 인터 예측(움직임 보상)은 H.264/AVC 디코더 가운데서 병목현상을 일으키는 가장 주요한 부분이다.

인터 예측의 성능 향상을 위하여 픽셀 보간 블록[3]에 대한 성능 향상에 초점을 맞출 수도 있고 메모리 액세스 구조[4]에 대한 성능 향상에 초점을 맞추어 전체 인터 예측 블록에 대한 성능을 향상 시킬 수 있는데 본 논문에서는 인터 예측 가운데서 참조 블록을 가져 오기 위한 메모리 패취 사이클 수를 줄이는데 초점을 맞추었다.

제안된 디자인은 그림 1과 같이 Motion Vector Decoder 블록, Cycle Controller 블록, Sliding Windows 블록, Interpolator 블록, Summation 블록과 외부 메모리로 구성 되었다.

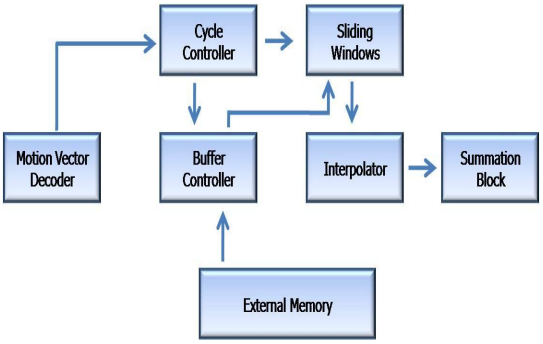


그림 1. 인터 예측 구조

Motion Vector Decoder에서는 MVD를 이용하여 MV를 계산하고 Cycle Controller에 전송한다. Cycle Controller에서는 계산된 MV와 블록 타입에 근거하여 메모리 패취 사이클을 계산한다. Buffer Controller에서는 계산된 사이클에 근거하여 메모리에서 참조 픽셀을 패취하고 Sliding Windows에서는 픽셀 보간에 필요한 수직, 수평 픽셀을 할당 받고 Interpolator에서는 휘도 블록과 색차 블록에 대한 픽셀 보간을 진행한다.

본 논문은 아래와 같이 구성 되었다. 2장에서는 기존의 인터 예측 구조에 대해 설명하고 3장에서는 새로운 인터 예측 구조에 대해 설명한다. 4장에서는 제안한 구조와 기존 구조와의 성능비교를 하고 5장에서 결론을 맺는다.

II. 기존 설계의 인터 예측 알고리즘

H.264/AVC 디코더는 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4 등 다양한 블록 타입을 지원하는데 레퍼런스 소프트웨어(JM)에서는 중복 픽셀에 대해 제거 하지 않고 항상 4x4 블록에 대한 9x9 참조 블록을 패취하게 된다. 그림 2는 레퍼런스 소프트웨어에서 블록 타입이 8x8인 경우를 예로 든 것이다.

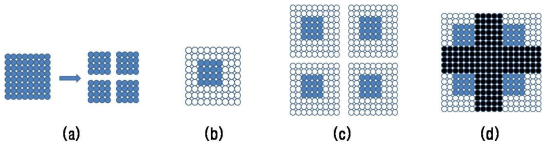


그림 2. 8x8블록 타입의 문제점

블록 타입이 8x8일 경우 4개의 4x4(a)블록으로 나뉘는데 하나의 4x4 블록에 대한 참조 블록 사이즈는 9x9(b)이다. 4개의 4x4 블록에 대한 9x9 참조 블록(c)을 메모리에서 패취하게 되면 (d)에서와 같이 크로스로 표시된 중복 픽셀까지도 패취한다. 불필요한 중복 픽셀을 패취하면 추가적인 메모리 패취 사이클이 소요 되는데 기존에 이미 설계된 디자인에서는 이러한 문제를 해결하기 위하여 8x8 블록 타입과 4x4 블록 타입을 고려하여 설계하였다. 블록 사이즈가 8x8보다 크거나 같을 경우, 여러 개의 8x8 블록으로 나뉘어서 처리되고 블록 사이즈가 8x8보다 작을 경우, 여러 개의 4x4 블록으로 나뉘어서 처리 되어 일부 중복 픽셀을 줄여 참조 픽셀 패취 사이클 수를 줄였다.

III. 제안된 디자인 구조

제안된 디자인에서는 더 많은 중복 픽셀을 줄여 참조 픽셀 패취 사이클 수를 줄이기 위하여 8x8 블록 타입과 4x4 블록 타입만이 아닌 여러 가지 블록 타입의 참조 픽셀 패취 사이클을 고려한 구조를 설계하였다.

3.1 Cycle Controller

Cycle Controller는 Motion Vector Decoder에서 디코딩된 MV와 블록 타입에 의하여 참조 픽셀 패취 사이클을 계산한다. 그림 3은 휘도 블록에서의 정수 픽셀, 1/2 픽셀, 1/4 픽셀 위치를 나타 낸다.

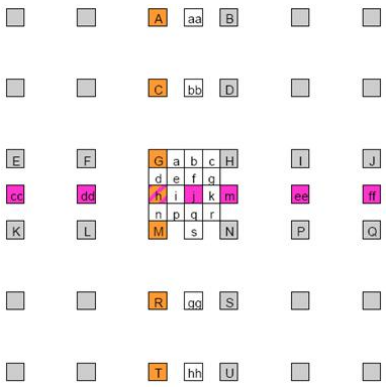


그림 3. 휘도 블록 픽셀 위치

정수 픽셀 사이에 1/2 픽셀, 1/4 픽셀 위

치가 존재 하는데 그 위치는 각각 mvx[1:0]과 mvy[1:0]에 의해 결정 된다.

그림 4는 16x16 블록 타입일 경우 각 픽셀 위치에서의 패취 해야 할 참조 블록의 크기를 보여 준다.

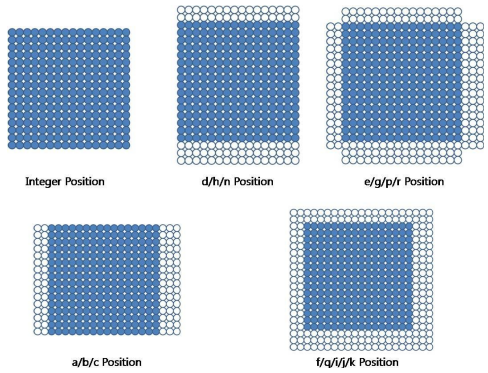


그림 4. 16x16블록 타입의 참조 블록

정수 위치일 때, mvx[3:2]=00 일 경우 하나의 수평라인을 패취 하는데 필요한 사이클 수는 32bit 데이터 폭을 가진 메모리를 기준으로 4 사이클이 필요하고 mvx[3:2]!=00 일 경우 5 사이클이 필요하다. 따라서 전체 16x16 사이즈인 참조 픽셀을 패취 하는데 64 혹은 80 사이클이 필요하다. 마찬가지로 d/h/n 위치에서는 84 혹은 106 사이클, e/g/p/r 위치에서는 122 사이클, a/b/c 위치에서는 96 사이클, f/q/i/j/k 위치에서는 126 사이클이 필요하다. 계산된 사이클 수는 Buffer Controller에 전송된다.

3.2 Buffer Controller

Buffer Controller에서는 외부 메모리로부터 참조 픽셀을 패취 한다. 할당된 사이클 수와 MV에 의하여 참조 픽셀을 패취하여 버퍼에 저장한다. 그림 5는 블록 타입이 16x16, 8x8 일 때 참조 픽셀이 버퍼에 저장 되는 위치를 보여 준다.

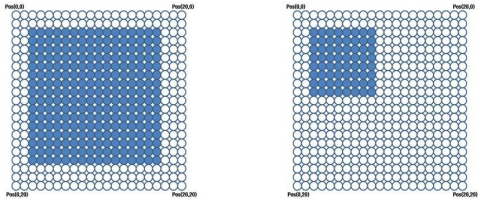


그림 5. 16x16, 8x8 블록 타입의 일 때 참조 픽셀이 버퍼에 저장되는 위치

그림 5에서 mvx[1:0]=00, mvy[1:0]=00 일 경우인데, 즉 픽셀 위치가 정수 픽셀일 때 두 블록에 대한 픽셀 보간을 하기 위하여 각각 16x16, 8x8 참조 블록이 필요한데 16x16일 경우 버퍼의 위치 (2,2)로부터 (17,17)까지의 위치에 저장 되고 8x8일 경우 버퍼의 위치 (2,2)로부터 (9,9)까지의 위치에 저장 된다. 그러므로 버퍼는 최대 16x16에 대한 참조 블록 픽셀을 저장 하기위하여 21x21개의 8 비트 레지스터로 구성 된다.

3.3 Sliding Windows

Sliding Windows는 수평 윈도우와 수직 윈도우로 구성 되었는데 9개의 수평 윈도우와 4개의 수직 윈도우로 구성되었다. 그림 6은 픽셀 보간이 4사이클 필요할 때 Sliding Windows를 보여 준다.

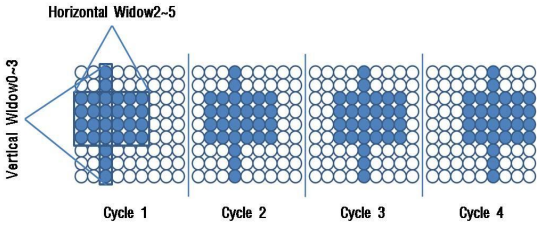
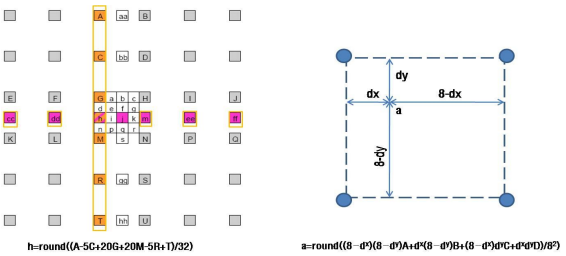


그림 6. 픽셀 보간이 4 사이클인 슬라이딩 윈도우

픽셀 보간은 4x4 블록에 대하여 계산을 하는데 j/f/q 위치일 때 5 사이클, i/k 위치일 때 8 사이클, 나머지 경우는 4 사이클이 필요하다. 그림 6에서와 같이 픽셀 보간은 수직, 수평 위치에서 동시에 진행되기 때문에 픽셀 보간이 4 사이클일 경우 4개의 수평 슬라이딩 윈도우와 4개의 수직 슬라이딩 윈도우에서 동시에 참조 블록을 저장한다.

3.4 Interpolator

픽셀 보간은 휘도 블록 보간과 색차 블록 보간으로 나뉘는데 그림 7은 표준에서 정의 된 픽셀 보간의 공식을 보여 준다.



(a) (b)

그림 7. 휘도 블록, 색차 블록 계산 공식

그림 7의 (a)공식은 휘도 블록에 대한 1/2 위치 픽셀 보간 공식이고 (b)공식은 색차 블록에 대한 픽셀 보간 공식이다.

그림 8은 휘도 블록에 대한 Interpolator 하드웨어 구조 이다.

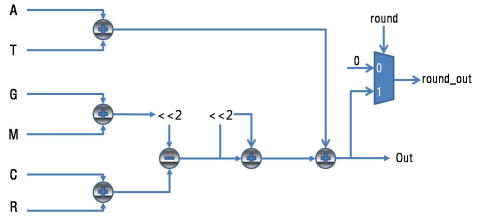


그림 8. 휘도 블록 Interpolator

그림 8에서 보다 시피 휘도 블록 한 픽셀에 대한 픽셀 보간을 진행하는데 1 사이클이 필요한데 수평 휘도 Interpolator는 9개, 수직 휘도 Interpolator는 4개가 존재 하는데 수직, 수평 위치에서 동시에 픽셀 보간을 진행한다. 그러므로 동일한 사이클 동안 수직위치에서의 4개의 픽셀 위치가 계산 된다.

그림 9는 색차 블록 픽셀 보간 구조이다.

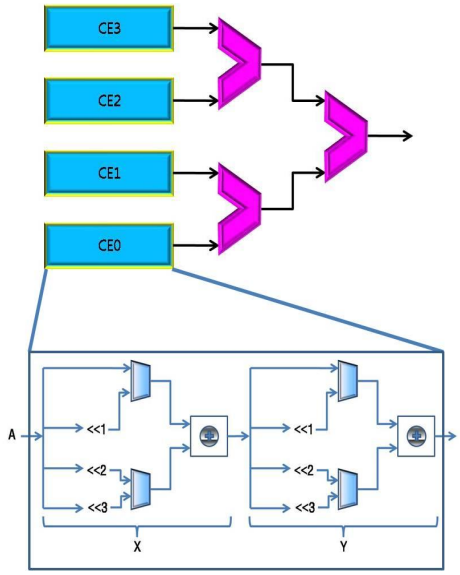


그림 9. 색차 블록 Interpolator

색차 블록 Interpolator는 4개의 Interpolator가 동시에 실행되는 병열구조로 되어 있는데 한 사이클 내에 2x2 블록에 대한 픽셀 보간을 진행하여 4x4 블록을 보간 하는데 4 사이클이 소요된다.

Ⅳ. 성능 비교

기존 설계는 8x8 블록과 4x4 블록에 대한 참조 블록을 고려하였고 제안된 설계에서는 다양한 블록 타입에 대한 참조 블록을 고려하여 설계하였다. 표 1은 기존 설계와 제안된 설계의 성능 비교 표이다.

표 1. 기존 설계와 제안된 설계의 성능 비교

테스트 파일	highway	coastguard	foreman	mother-daughter	trevor
프레임 수	300	300	300	300	150
인터 MB 수	29299	29442	28920	29540	14077
기존 설계에서의 휘도 블록 패취 사이클	2954324	4019073	4407679	2734982	1547479
제안된 설계에서의 휘도 블록 패취 사이클	2555928	3327067	3587541	2467449	1395812
기존 설계에서 하나의 MB에 대한 패취 사이클	100.8	136.5	152.4	92.6	109.9
제안된 설계에서 하나의 MB에 대한 패취 사이클	87.2	113.0	124.1	83.5	99.1
패취 사이클 감소율(%)	13.5	17.3	18.6	9.8	9.8

표 1에서 보다 시피 150~300 프레임인 5개의 입력 파일을 사용하여 실험을 진행하여 참조 블록 패취 사이클을 줄인 결과를 확인 하였는데 최소 9.8%, 최대 18.6%의 참조 블록 패취 사이클 감소를 가져 왔다.

Ⅴ. 결 론

본 논문에서는 H.264/AVC 디코더의 인터 예측 블록에 대한 성능을 향상시키기 위한 하드웨어 구조를 제안 하였다. 기존 하드웨어 구조와 달리 다양한 블록 타입에 대해 고려하고 그에 대한 참조 픽셀 메모리 패취 사이클을 줄이는 하드웨어 구조를 제안 하였다. 실험 결과 제안된 구조는 기존 구조와 비교 할 때 최소 9.8%, 최대 18.6%의 성능향상을 가져 왔다.

참고문헌

[1] Joint Video Team (JVT) of ISO/IEC & ITU-T VCEG,, H.264 : Advanced video coding for generic audiovisual services, March, 2005

[2] 김대연, 임성창, 이영렬, “H.264/AVC의 화면내 예측을 위한 새로운 고속 모드 결정 방법”, 한국방송공학회학술대회, pp. 117-120, 2006년 11월.

[3] Sheng-Zen Wang, Ting-An Lin, Tsu-Ming Liu, Chen-Yi Lee, “A new motion compensation design for H.264/AVC decoder”, ISCAS 2005. IEEE International Symposium, 23th~26th May, 2005.

[4] Ronggang Wang, Mo Li, Jintao Li, Yongdong Zhang, “High throughput and low memory access sub-pixel interpolation architecture for H.264/AVC HDTV decoder”, Consumer Electronics, IEEE Transactions Volume 51, pp. 1006-1013, August, 2005.