H.264/AVC에 적용 가능한 고속 deblocking 필터 연구

정덕영*, 김원삼, 손승일

한신대학교

A study on the fast deblocking filter for H.264/AVC

Duck Young Jung*, Won Sam Kim, Seung Il Sonh

Hanshin university

E-mail : mykor01@naver.com

요 약

동영상과 관련된 멀티미디어가 많은 관심을 받으며 영상 압축 기술에 대한 관심이 높아지고 있는 가운데, 최근 다른 표준보다 두 배 이상 좋은 새로운 비디오 코딩 표준인 H.264/AVC의 압축 기술이 발표되었다. 이 기술은 지상파 DMB와 PMP, 카메라폰 그리고 핸드폰의 게임과 음악 및 영상에 관련된 컨텐츠에서 고품질의 영상을 보다 효율적으로 제공 한다. 이에 본 논문에서는 H.264/AVC의 부호화 과정에서 발생하는 오류로 인한 블록화를 최소화하기 위해 사용되는 deblocking 필터의 메모리와 처리속도의 향상을 제안하였다. 27*32SRAM을 사용하여 Vertical edge를 모두 처리하고 Horizontal edge를 처리하는 방식이 아닌 한 블록에 대한 Vertical edge후에 바로 Horizontal edge를 처리함으로써 28(prebuffering)+96(Y)+32(Cb)+32(Cr)=188clocks에 16x16 블록 처리가 완료되는 deblocking 필터를 제안하여 하드웨어 설계언어인 VHDL언어로 설계하였다. 그리고 FPGA칩인 XCV1000E에 다운로드하여 칩 레벨의 시뮬레이션을 수행함으로써 설계된 deblocking 필터를 검증하였다.

Ⅰ. 서 론

최근 동영상과 관련된 멀티미디어가 많은 관심을 받으면서 제안된 채널과 데이터양 그리고 처리속도에 대한 문제점들이 제기되어 보다 적은 데이터를 빠르게 처리할 수 있는 코딩 방식이 발표되었다. 이렇게 처리할 경우 발생하는 오류로 인해 블록화가 발생하게 되어 적은 데이터로 빠르게 보낸다 해도 만족스러운 화질의 압축된 영상을 얻기 힘들기 때문에 효율적인 필터기법이 요구된다. 그래서 MPEG-4 표준의 경우에는 DCT와 양자화 과정을 통하여 발생하는 블록화 현상을 효율적으로 처리하기 위해 후처리로서 부호화 과정 및 복호화 과정과 무관하게 복호화 된 영상에 대해 선택적으로 필터링을 한다.

하지만, 이보다 두 배 이상의 압축 코딩 표준인 H.264/AVC는 주관적 화질의 개선과 부호화 효율의 개선을 위하여, 부호기/복호기 내부에 루프 필터를 위치시킨다. H.264/AVC의 부호화 알고리즘은 4x4블록 단위의 부호화 및 공간 예측 방식을 사용하므로, 양자화된 변환계수의 분포가 기존의 표준과 다르게 표현된다. 따라서 각 블록의 부호화 형태, 양자화 크기 및 intra와 inter 등을 고려하여 필터링을 수행는 deblocking필터가 우수한 성능을 나타낸다. [1].

Ⅱ. deblocking 필터

H.264는 4x4 블록을 기본으로 DCT와 양자화를 통한 transformation하고, motion compensation에 대해서도 역시 4x4의 작은 블록 사이즈로 된다. 그런 까닭에 deblocking 필터는 우선 4x4 블록사이의 boundaries를 체크한다. deblocking한 macroblock을 하나씩 raster scan order를 마친다.

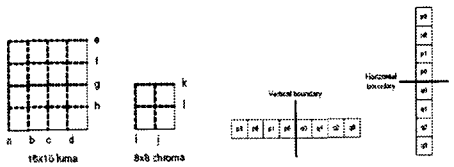

그림 1. macroblock의 수직과 수평에 대한 순서도

서로 다른 macroblock으로 처음엔 휘도 성분의 4개의 horizontal 경계를 필터처리하고, 그 다음에 휘도 성분의 4개의 vertical 경계를 필터 처리한다. 그리고 각 색차 성분 2개를 같은 방식으로 필터 처리한다. 각각의 인접한 4x4luma블록들 사이에 boundary의 boundary strength(BS) 할당과 수행 순서는 그림1에서 보여준다.

필터링은(슬라이스 경계의 가장자리를 제외한)

매크로 블록 내의 4x4 블록의 수직 또는 수평 가장자리에 대해 그림1과 같은 순서로 필터링 한다. 필터링 과정의 선택은 경계 세기(boundary strength)와 경계 주위의 이미지 샘플 변화에 의해 좌우된다. 경계 세기 파라미터(bs)는 그림2에 보여주는 것과 같이 5단계 중에서 선택된다. 그림1은 인접한 블록 q에 있는 수직 또는 수평 경계 양쪽의 4개의 샘플(p0,p1,p2,p3와 q0,q1,q2,q3)을 나타내고 있다.[1-2].

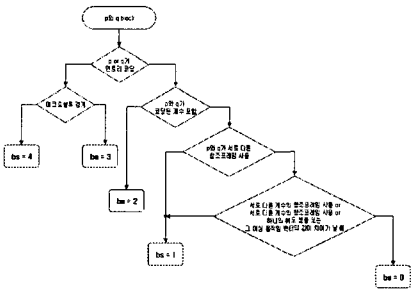

그림 2. boundary strength를 결정하는 흐름도

p0,p1,p2,p3 와 q0,q1,q2,q3로부터의 샘플 그룹들은 다음과 같을 때만 필터링 한다.

α와 β는 표준안에서 정의된 임계값으로 두 개의 블록 p와 q의 양자화 파라미터 QP의 평균에 따라 증가하며, 필터링의 결정의 효과는 원본이미지의 블록 경계 주위에 많은 변화가 있으면 필터링하지 않는다.

$$bs > 0,\ |p0 - q0| < \alpha,\ |p1-p0| < \beta,\ |q1-q0| < \beta \quad (1)$$

만약 식(1)에 만족하면 bs∈{1,2,3}와 bs=4일 경우에 따라 필터링한다. bs∈{1,2,3}일 경우, 4-tap 필터가 입력 p1,p0,q0,q1에 대해 적용되어 필터링된 출력 p'0과 q'0을 생성한다. 만약 휘도 이면서 |p2-p0|<β일 경우 또 다른 4-tap 필터가 입력 p2,p1,p0,q0에 적용되어 필터링 된 출력 p'1이 생성된다. q에 관하여도 동일하게 적용된다. bs=4일 경우, 휘도 블록이거나 |p2-p0|<β이고 |p0-q0|<((α>>2)+2)을 때 5-tap 필터가 p'0과 p'2를 생성하고 4-tap 필터가 p'1을 생성하며, 위 조건에 만족하지 않을 경우 3-tap 필터가 p'0을 생성한다[3-4].

III. deblocking 필터 설계

3.1 deblocking 필터 블록도

그림3은 deblocking 필터의 입출력 신호도를 보여주고 있다. 입력포트를 통하여 32bits의 현재 블록 데이터와 QP가 reconfigurable 1D 필터의 horizontal 필터에 입력되면 control1의 제어신호를 통해서 그림4의 1,2,3,4,5는 처리하지 않고 전송하며 6번부터 처리하여 4x32bit buffer1,2,3에 저장하고 저장된 데이터를 이용하여 vertical 필터에서 연산을 수행한다.

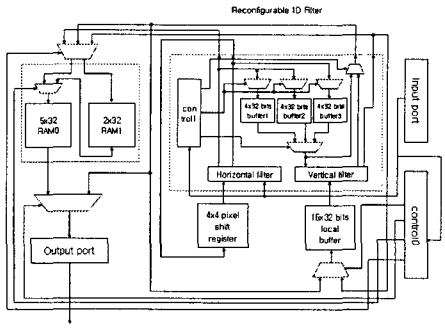

그림 3. 입출력 신호도

순차적으로 데이터를 연산하기 위해 4x4 shift register를 통해 전 pixel을 저장하고 다음 연산에 출력하여 horizontal 필터수행 시 입력으로 사용하며, vertical 필터를 수행하기 위해 horizontal 필터에서 처리한 데이터를 16x32 bits local buffer에 저장하고 vertical 필터 수행 시 입력으로 사용하여 deblocking 필터 처리 시 발생하는 지연시간을 최소화 하여 처리한다.

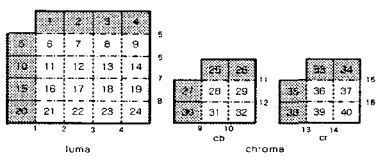

그림 4. deblocking 필터 입력 순서 및 처리 순서

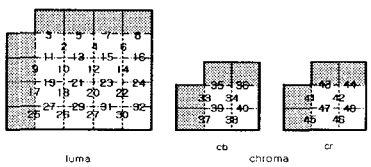

그림 5. 제안된 2-D processing 처리 순서

또한 deblocking 필터 처리가 완료된 데이터를 순차적으로 처리하기 위해 5x32 sequence ram, 2x32 sequence ram에 control0의 제어신호로 저장하고 처리하여 최종적으로 28 clock의 초기화 후부터 순차적으로 데이터를 전송한다. 이와 같은 처리는 그림5에서 보여주고 있다[5-6].

3.2 horizontal 필터 and vertical 필터

horizontal 필터와 vertical 필터를 수행 할 때 공통적으로 bs_select와 bs=4 그리고 bs∈{1,2,3}과 α,β,△ 등의 필터ing에 필요한 값을 구하는 부분으로 구성 되어있으며, 4x4 pixel shift register에서 전송되는 이전 블록(p)과 input port에서 전송받은 현재 블록 데이터(q)를 이용하여 bs를 구하고 그에 따른 α,β,△ 등을 연산한

다. 연산된 결과를 이용하여 그림6와 그림7의 table을 이용하여 각각의 값을 연산한다.

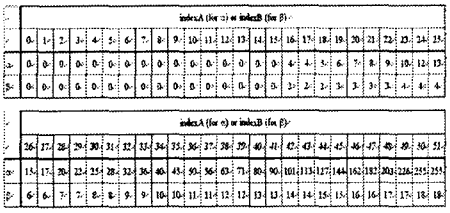

그림 6. α, β table

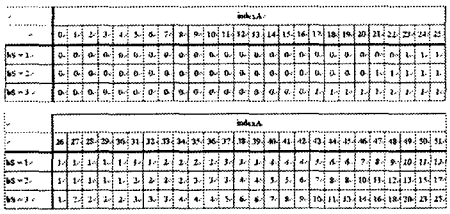

그림 7. △ 연산에 사용되는 tc0 table

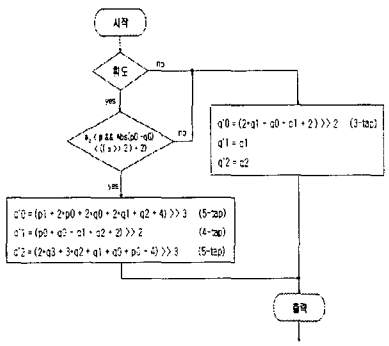

그림 8. bs=4 처리 흐름도

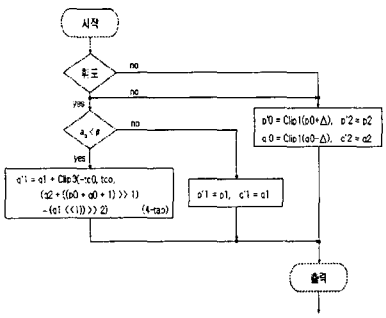

그림 9. bs∈{1,2,3} 처리 흐름도

연산된 q'0,q'1,q'2 값은 4x32 bits buffer1,2,3에 저장하고 p'0,p'1,p'2는 4x4 pixel shift register에 저장하여 지연을 최소화한 연산 처리를 수행한다. 그림8과 그림9는 bs=4와 bs={1,2,3}의 처리 흐름을 보여주고 있다[5-6]. horizontal 필터에서는 4x4 bits buffer1,2를 통하여 horizontal 필터 수행시 transforse된 데이터와 16x32 bits local buffer에서 전송된 데이터를 이용하여 horizontal 필터가 처음 처리하고 8clock 후에 horizontal 필터와 같은 수행을 처리한다[7-8].

3.3 deblocking 필터에서 사용되는 메모리

그림3에서 보여주는 것과 같이 deblocking 필터에서 사용되는 메모리는 크게 4x32 bits buffer1,2,3과 4x4 shift register, 16x32 local buffer, 5x32 그리고 2x32로 구성된다. 각각의 처리는 다음과 같다. 4x32 bits buffer1은 horizontal 필터 수행한 후 처리된 결과를 vertical 필터 수행을 위해 저장하고, vertical 필터 수행 시 transpose하여 전송하고 동시에 4x32 bits buffer2에서도 동일하게 수행한다.

4x32 bits buffer3은 vertical 필터 수행 시 동시에 발생하는 출력을 저장한 후 transpose하여 전송한다. 5x32 bits buffer and 2x32 bits buffer는 필터 수행 완료된 데이터를 순차적으로 처리하기 위한 저장 공간으로써 5, 10, 15, 20, 21, 22, 23, 24, 27, 30, 35, 38과 9, 14, 19, 24, 29, 32, 37, 40을 저장하여 순차처리 한다. 그리고 16x32 bits local buffer는 vertical 필터 처리된 데이터를 저장하여 다음 vertical 필터 처리 시 p 데이터로 사용한다.

IV. 설계 결과

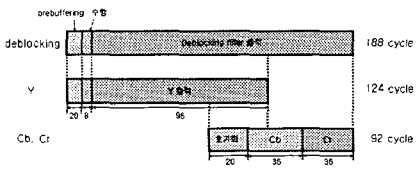

그림 10. deblocking 필터 수행 cycle

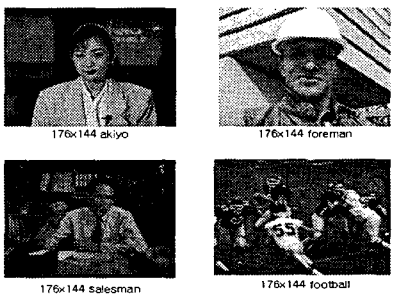

그림 11. 176*144 qcif 입력 영상

표1. memory size와 Design 비교

Design / 항목	Proposed (single port SRAM)	[3] Proposed (single port SRAM)	[8] Basic type (single port SRAM)	[8] Advanced type (dual port SRAM)	[8] Basic type (two port SRAM)	[8] dual array type (two port SRAM)
Cycle per MB	188	336	876	814	782	614
Memory size	27x32bits	80x32bits	160x32bits			

그림10은 16x16 블록을 deblocking 필터를 수

행하여 처리한 cycle을 보여주고 있다. 한 블록에 대한 vertical 필터를 모두 처리한 후에 바로 horizontal 필터를 처리함으로써 16x16 블록을 28(prebuffering)+96(Y)+32(Cb) +32(Cr)=188clocks로 처리하였다. 그림11는 deblocking 필터에서 사용된 176*144의 입력 영상을 보여주고 있다.

본 논문에서 제안하여 설계한 deblocking 필터의 수행 cycle과 memory에 대해 다른 논문의 cycle과 memory를 비교하였다. 그 결과 기존 논문들보다 2배에서 약4.5배 정도의 cycle이 감소하였고 Memory size는 3에서 약6배정도 적은 Memory를 사용하였고, 그 분석결과는 표1에서 보여주고 있으며 본 논문에서 사용한 Memory는 표2에서 보여주고 있다.

표2. 제안한 memory size

Design		Proposed	
Memory size	local buffer	16x32 bits	27x32 bits
	Shift register	4x32 bits	
	Ram0	5x32 bits	
	Ram1	2x32 bits	

V. 결론

deblocking 필터에서 사용되는 메모리를 27*32bits를 사용하여 설계함으로써 칩 사이즈를 줄였고, 16x16블록을 deblocking 필터 처리 하는데 28(prebuffering) + 96(Y) + 32(Cb) +32(Cr) = 188clocks이 요구 되었다. 이는 메모리 사이즈 줄여서 설계하여 192(Y) + 72(Cb) + 72(Cr) = 336clocks로 처리한 기존 논문[3][8]과 비교했을 때 상당히 빠르고 적은 메모리를 사용했음을 알 수 있다.

본 논문에서 설계한 deblocking 필터는 적은 사이즈와 빠른 처리속도로 처리하므로 실시간으로 방송하는 지상파 DMB용 핸드폰이나 PMP 그리고 각종 핸드폰에 관련된 컨텐츠에 사용 가능하다.

참고문헌

[1] Lain E.G. Richardson, "H.264 and MPEG-4", 홍릉과학출판사, 2004.9

[2] Joint Video Team of ISO/IEC MPEG & ITU-T VCEG, "JVT-G050r1", 2003.

[3] Chao-Chung Cheng, Tian-Sheuan Chang, "An Hardware Efficient Deblocking Filter for H.264/AVC", IEEE 2003. pp235-236

[4] H.264/AVC reference software JM7.2, jul. 2003.

[5] Bin Sheng, Wen Gao, Di Wu, " An Implemented architecture of deblocking filter for H.264/AVC", IEEE 2004 ICIP. pp665-668.

[6] John Wiley · Sons, "Video CODEC Design", WILEY, 2002.

[7] Information Technology Advanced audio video code Part Two : Video (March 22, 2004).

[8] Y.-W. Huang, T.-W. Chen, B.-Y. Hsieh, T.-C.Wang, T.-H. Chang, and L.-G. Chen, "Architec-ture design for deblocking filter in H.264/JVT/AVC", in proceedings of 2003 IEEE Internatio-nal Conference on Multimedia and Expo (ICME 2003), Baltimore, USA, July.