논문 2008-45SD-1-10

효율적인 메모리 관리 구조를 갖는 H.264용 고성능 디블록킹 필터 설계

(Design of a Pipelined Deblocking Filter with efficient memory management for high performance H.264 decoders)

유 용 훈*, 이 찬 호**

(Yonghoon Yu and Chanho Lee)

요 약

고성능 영상 압축 알고리즘으로서 널리 사용되고 있는 H.264 디코더의 디블록킹(Deblocking) 필터는 복호된 영상의 블록화 현상을 제거함으로써 영상의 질을 높이는 역할을 하는데 연산량이 많은 유닛중 하나이다. 본 논문에서는 효율적인 디블록킹 필터 설계를 위해 파이프라인 구조 및 1-D 필터를 사용하고 효율적인 메모리 관리를 통해 하드웨어 면적과 연산 사이클 수를 줄이고 H.264 디코더의 성능을 향상시킬 수 있는 하드웨어 구조를 제안한다. 제안된 구조에서는 픽셀의 재배치를 통해 동일한 1-D 필터를 이용하여 수직방향의 필터연산과 수평방향의 필터연산을 모두 지원한다. 또한 4 개의 메모리 블록 구조를 이용하여 현재 매크로블록의 픽셀과 인접한 다른 매크로블록의 픽셀의 접근 및 저장을 효율적으로 할 뿐만 아니라 필터 연산중에 움직임 보상기의 출력 픽셀을 저장하여 디블록킹 필터와 움직임 보상기 사이의 병목현상을 제거하였다. 이를 통해 디블록킹 필터에 관련된 메모리의 크기를 최소화하고 H.264 디코더의 성능을 향상시키는 이점을 얻을 수 있다. 제안된 디블록킹 필터는 Verilog-HDL을 이용하여 설계하고 FPGA를 통해 검증하였다. 합성 결과 77 MHz에서 HD 영상 디코딩이 가능함을 확인하였다.

Abstract

The H.264 standard is widely used due to the high compression rate and quality. The deblocking filter of the H.264 standard improves the quality of images by eliminating blocking artifacts of pictures, and it requires a lot of computation. We propose a new hardware architecture for the deblocking filter with pipelined architecture, 1-D filters which support both horizontal and vertical filtering and efficient memory management. Four memory blocks are configured for the efficient storage and access of the current macroblock and adjacent referenced sub-macroblocks, and the pixel data from the motion compensation unit can be transferred without waiting during the computation cycles of the deblocking filter. The number of computation cycles and the hardware area are reduced using the proposed architecture, and the performance of the H.264 decoder is improved. We design the deblocking filter using Verilog-HDL and implement using an FPGA. The designed deblocking filter can be used for decoding HD quality images at 77 MHz.

Keywords : H.264, Deblocking filter, Memory management, Pipelined architecture, 1-D filter

I. 서 론

멀티미디어 데이터 처리 기술의 발달로 사용자의 눈높이가 높아짐에 따라 점차 고화질의 영상을 요구하게 되었다. 그러나 기존의 영상 압축 알고리즘으로는 화질이 향상되면 저장 용량이 늘어나는 문제점이 있고 휴대용 기기의 경우에는 저장 공간의 제한으로 고화질의 영

* 학생회원, 숭실대학교 전자공학과
(Dept. of Electronic Engr., Soongsil University)
** 정회원, 숭실대학교 정보통신전자공학부
(School of Electronic Engr., Soongsil University)
※ 본 논문은 산업자원부가 지원하는 국가 반도체연구개발사업인 시스템집적반도체기반기술개발사업 (시스템IC2010)을 통해 개발된 결과임을 밝힙니다. 또한 IDEC의 CAD 툴 지원을 받았습니다.
접수일자: 2007년7월16일, 수정완료일: 2008년1월14일

상을 제공하기 어렵다. 이러한 요구 조건을 만족시키기 위해 고성능 영상 압축 알고리즘의 필요성이 대두되었고 이에 따라 H.264 (또는 MPEG4/AVC) 압축 영상 알고리즘이 개발되었다. H.264 압축 표준에서는 4*4 블록 단위의 움직임 보상(MC) 및 변환, 양자화, 디블록킹 필터(DF), 가변길이코드(VLC) 등을 사용하여 기존의 영상 압축 알고리즘보다 뛰어난 압축률과 고화질을 제공한다.

H.264 압축 표준도 기존의 영상 압축 알고리즘과 유사하게 매크로 블록을 기반으로 연산을 수행한다. 매크로 블록을 기반으로 연산을 하면 블록간의 오차가 생기기 마련이다. 블록간의 오차가 심하면 영상은 마치 모자이크처럼 보일 수도 있다. 이런 블록간의 오차를 블록화 현상이라 한다. 블록화 현상은 영상의 화질이 떨어지는 문제점을 발생시킨다. 특히, 다음 프레임 움직임 보상 연산 시 현재 영상을 참조하여 연산을 하는데 한 번 저하된 영상의 화질은 블록화 현상이 존재하는 상태로 계속 참조되어 다음 영상을 복원할 때 블록화 현상이 계속해서 누적된다. 이러한 영상의 화질 문제를 해결하기 위해서 H.264에서는 복원된 영상에 필터연산을 가함으로써 블록화 문제점을 해결함과 동시에 복원된 영상이 움직임 보상 연산 시 참조 프레임으로 이용되어, 이후 복호되는 영상의 화질도 향상시킨다. 또한 같은 화질의 영상을 전송 할 경우 기존 영상 압축 알고리즘과 비교해 본다면 비트 전송률도 5 - 10% 줄이는 이점을 얻을 수 있다[1].

한 매크로 블록 연산 시 DF는 16 x 16 휘도 성분과 8 x 8 색차 성분 2 개가 필요하고, 매크로 블록 경계에서도 매크로 블록의 약 절반 크기에 해당하는 참조 매크로 블록까지 필요하게 되어 상당한 크기의 메모리가 요구된다. 메모리에 저장된 픽셀은 수평경계와 수직경계를 기준으로 두 번 연산됨으로 픽셀의 저장과 불러오기가 빈번하게 행해진다. 이 때 필요한 픽셀의 접근과 관리를 체계적으로 하는 것이 중요하다. 이와 더불어 DF에는 3, 4, 5 탭(tap)의 필터가 존재하는데 DF 하드웨어 설계 시 다양한 탭의 필터의 효율적인 구현과 필요한 메모리의 배치 및 관리를 어떻게 할 것인지 중요한 요소가 된다.

기존에는 DF 설계 시 매크로 블록의 모든 픽셀 값과 참조 픽셀 값을 모두 저장한 뒤에 필터연산을 시작하였다[1~3]. 이 경우에 필요로 하는 8 bit 픽셀은 총 640개로 32 bit 메모리에 저장한다고 가정한다면 160 x 32 bit의 메모리가 필요하게 된다. 또한 휘도성분과 색차성분을 모두 저장한 뒤 필터연산을 할 경우 MC의 출력 픽셀을 DF에 저장 할 수 없다. 이 경우 DF와 MC사이에 FIFO 및 메모리를 사용하거나 MC 내부에 메모리를 사용하여 해결하기도 하지만 추가 메모리가 필요하다는 단점이 있다. 버퍼 메모리가 없을 경우에 DF가 연산 중에 데이터를 받을 수 없으면 MC가 멈추는 병목현상이 나타난다.

본 논문은 DF에서 필요로 하는 내부 메모리를 4 개의 블록으로 나눠서 DF와 MC 사이 또는 MC 내부에 별도의 메모리를 두지 않고 DF와 MC 간의 병목현상을 제거하여 H.264 디코더의 성능을 향상시키는 구조를 제안한다. 또한 파이프라인 구조를 사용하여 동작 주파수를 향상시키고, 1-D 필터를 이용하여 수직방향과 수평방향에 이용함으로써 DF의 필터수를 줄였다. 또한 1-D 필터를 통과한 픽셀을 반대방향으로 재배치함으로써 수평경계 필터연산 시 효율적인 메모리 접근이 가능하도록 하여 연산 사이클 수를 줄였다.

Ⅱ. 디블록킹 필터 구조

1. DF 알고리즘

H.264 영상 압축 알고리즘은 영상을 복호할 때 4*4 서브 매크로 블록을 기반으로 연산을 한다. 블록 기반의 압축 알고리즘에서는 복호 과정이 블록 단위로 진행되므로 각 서브 매크로 블록 경계에서 블록화 현상이 발생하게 된다. H.264에서는 DF에서 블록화 현상을 제거하고 그 결과 영상의 질을 향상시켰다. 특히 블록화 현상이 두드러지게 나타나는 부분은 매크로 블록 경계이며 이를 해결하기 위해 강력한 필터연산으로 블록화 현상을 제거한다. 상대적으로 블록화 현상이 심하지 않을 경우 상대적으로 약한 필터연산을 하여 블록화 현상을 제거한다.

DF의 연산순서는 매크로 블록 내에서 수직경계 필터연산을 마친 뒤 수평경계 필터연산을 한다. DF의 연산순서는 그림 1의 점선을 기준으로 진행되며, 각 점선의 알파벳순으로 연산이 진행되는데 각 점선을 기준으로 양쪽 4 개씩의 픽셀을 취해 필터연산을 한다. 이때 필터연산에 직접적으로 영향을 주는 요소는 Bs(boundary strength) 계수, α, β, t_{C0}의 임계값이다.

Bs 계수는 블록간의 연관성을 기초로 필터의 강도를 정해주는 역할을 하는데 Bs 계수 연산 알고리즘이 그림 2에 나타나 있다. 인트라 모드의 매크로 블록 경계에서는 가장 강력한 필터연산을 하는 strong 모드이며

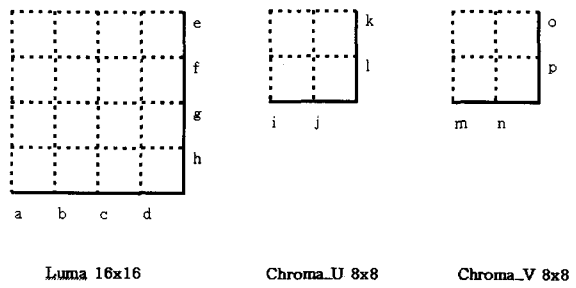


그림 1. 매크로블록의 수직경계와 수평경계의 연산 순서

Fig. 1. Order of filtering on vertical and horizontal edges in macroblocks.

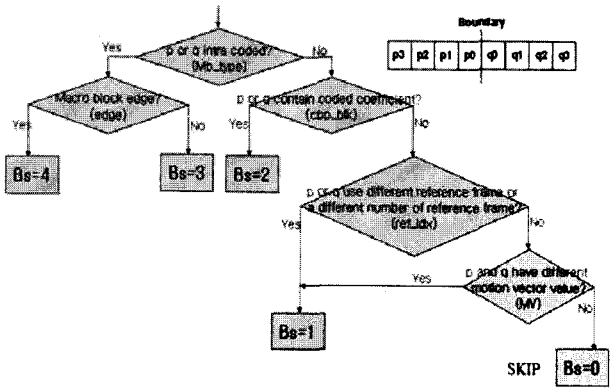


그림 2. Bs 계수 연산 알고리즘

Fig. 2. Calculation algorithm of the Bs coefficients.

이때 Bs 계수는 4가 된다. 인트라 모드 매크로 블록 경계가 아닐 때는 Bs 계수가 3이 되며 인터 모드일 경우 코딩 계수가 있으면 Bs 계수는 2가 되고 그렇지 않으면, 현재 픽셀과 참조 픽셀 p 와 q가 서로 다른 참조 프레임을 사용하거나 같은 프레임 내에서 다른 블록을 참조 하는 경우, 또는 서로 다른 움직임 벡터(MV)값을 가지게 되면 Bs 계수는 1이 된다. Bs 계수가 1부터 3까지 일 경우 normal 모드가 되며 일반적인 필터연산을 하게 된다. Bs 계수가 0인 경우에는 필터 연산이 필요 없는 경우로 필터연산을 하지 않는다. Bs 계수 연산에 필요한 파라미터들이 서브 매크로 블록 단위로 변하므로 Bs 계수도 그에 따라 계산된다.

Bs 계수뿐만 아니라 α, β, t_{C0}의 임계값도 필터연산에 영향을 준다. 각 임계값들은 H.264의 표준안에 정의되어 있으며 α, β의 값은 QP(Quantization Parameter)에 의해, t_{C0}의 값은 QP와 Bs 계수에 의해 정의된다[4].

2. DF 하드웨어 구조

본 논문에서 제안하는 DF의 하드웨어 구조는 1-D

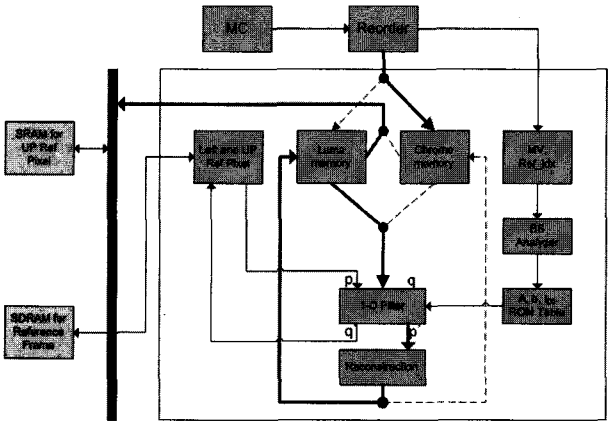


그림 3. 제안된 디블록킹 필터 구조

Fig. 3. Structure of the proposed deblocking filter.

필터, Bs 계산, 매크로 블록 메모리, α, β, t_{C0}의 ROM 테이블, 픽셀의 재배치(reconstruction) 유닛으로 구성되며, 그림 3의 블록 다이어그램이 나타나 있다. DF의 전체적인 연산순서를 살펴보면 Bs 계수를 계산하고 Bs 계수와 QP를 이용하여 α, β, t_{C0} 임계값을 ROM 테이블에서 읽은 뒤, Bs 계수, 임계값 그리고 픽셀 데이터를 이용하여 1-D 필터에서 필터연산을 진행한다. 필터연산이 끝난 픽셀은 재배치하여 메모리에 저장된다. 이러한 일련의 과정은 파이프라인 구조로 순차적으로 연산이 진행된다.

1-D 필터의 필터연산은 strong 모드와 normal 모드로 크게 나뉜다. Strong 모드의 경우 5, 4, 3 탭 필터를 사용하며 normal 모드의 경우 4 탭 필터를 사용한다. 모드뿐만 아니라 휘도와 색차의 구분에 따라 필터의 종류가 나뉘게 된다. 이렇게 다양한 나뉘는 필터를 각각 모두 구현할 경우 1-D 필터의 면적이 커지게 되므로 각 필터의 중복되는 연산은 다른 필터와 공유하는 방식으로 설계하였다. 즉 각 모드에 대응하여 필요한 필터의 수는 14 개인데 재사용 가능한 필터를 통합하여 11 개의 필터로 모든 필터링 작업을 수행한다. 따라서 파이프라인을 유지하는 범위에서 필터의 수를 21% 줄였다.

DF는 매크로블록 단위로 연산을 진행하여 많은 양의 메모리가 요구되어진다. 기존의 DF는 휘도와 2 개의 색차 픽셀을 모두 저장한 뒤에 연산이 진행된다[1~3]. 이 경우 많은 메모리가 필요하고 DF 연산 중에는 MC의 출력 픽셀을 저장할 수 없어 H.264 디코더의 파이프라인이 멈추는 병목현상이 나타날 수 있다. 이러한 단점을 개선하기 위해서 DF와 MC 사이에 FIFO나 메모리를 두어 해결하기도 하지만 추가 하드웨어가 필요하므

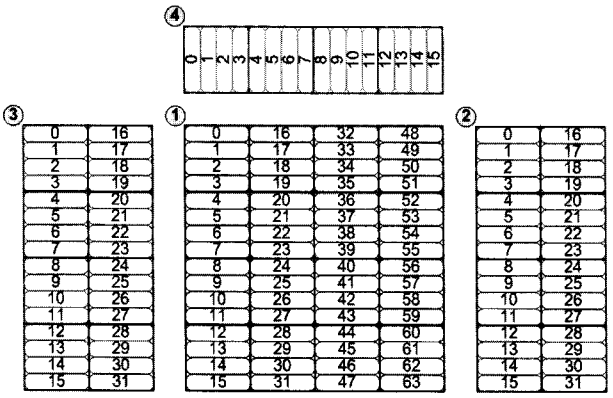


그림 4. 디블록킹 필터의 메모리 구조
Fig. 4. Memory organization of the proposed deblocking filter.

로 효율성이 감소된다. 이러한 단점을 보완한 메모리 블록 구조가 그림 4에 나타나 있다. ①번 메모리 블록은 매크로 블록의 휘도성분을 저장하고 ②번 메모리 블록은 매크로 블록의 2 개의 색차성분을 모두 저장한다. ①번 메모리 블록을 기준으로 왼쪽과 위쪽에 있는 ③, ④번 메모리 블록은 참조 매크로 블록을 저장하는데 쓰이며 휘도성분과 색차성분이 공유하여 사용한다.

여기서 ①번과 ②번 메모리는 DF 연산과 MC에서 출력되는 픽셀을 번갈아가며 저장하는 이중 버퍼링 역할을 한다. 먼저 ①번 메모리에 저장되어 있는 휘도 픽셀을 연산하는 동안 ②번 메모리는 MC에서 출력되는 색차 픽셀을 저장하며, 반대로 ②번 메모리에 저장되어 있는 색차 픽셀을 연산하는 동안 ①번 메모리는 MC에서 출력되는 휘도 픽셀을 저장한다. 즉, ①번과 ②번 메모리는 서로 분리되어 있고 DF 연산 유닛과 메모리 관리 유닛이 분리되어 독립적으로 동작하므로 휘도와 색차 연산이 별도로 진행되는 것을 이용하여 디코더 전체의 파이프라인을 유지하며 메모리 크기는 최소화 할 수 있다. 이에 대한 설명이 그림 3에 휘도와 색차 메모리를 중심으로 점선과 굵은 실선으로 표시되어 있다. 이렇게 메모리를 분리하여 사용함으로써 DF와 MC 사이의 병목현상을 제거 할 수 있다.

기존에 설계되었던 DF는 그림 1의 연산순서로 동작하였다[1~3]. 그러나 휘도와 2 개의 색차성분의 연산은 각각 독립적으로 진행되는 점을 이용한다면 기존의 연산순서를 바꿔도 필터연산의 결과에는 지장이 없다. 이러한 점을 이용하여 본 논문에서 제안하는 경계의 연산순서가 그림 5에 나타나 있다. 2 개의 색차성분에 대해 수직경계 방향으로 연산한 뒤 수평경계 방향의 연산을 진행한다면 휘도성분과 동일한 경계 연산 횟수가 되어

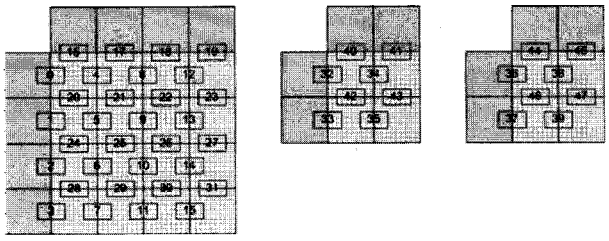


그림 5. 제안된 매크로블록의 수직경계와 수평경계 방향의 연산 순서
Fig. 5. Proposed order of filtering on vertical and horizontal edges in macroblocks.

제어유닛이 간단해진다. 또한 색차 성분 2 개를 함께 연산함으로써 두번 발생할 파이프라인 지연이 한번만 발생하게 하는 연산 사이클의 이점을 얻을 수 있다.

본 논문이 제안하는 DF 내부 메모리는 현재 매크로 블록의 휘도성분을 저장하는 64 x 32 bit 메모리와 색차성분을 저장하는 32 x 32 bit 메모리, 참조 매크로 블록을 저장하는 32 x 32 bit와 16 x 32 bit 메모리로 구성된다. ③번 32 x 32 bit 메모리는 현재 블록의 왼쪽 참조 매크로 블록을 저장한다. 0번에서 15번까지는 4 x 16 휘도 참조 매크로 블록을 저장하고 16번에서 31번까지는 2 개의 4 x 8 색차 참조 매크로 블록을 저장한다. ④번 16 x 32 bit 메모리는 현재 블록의 위쪽 참조 매크로 블록을 저장한다. 왼쪽 참조 매크로 블록이 위쪽 참조 매크로 블록보다 큰 이유는 영상의 복호 순서가 왼쪽에서 오른쪽으로 진행되기 때문에 다음 매크로 블록의 필터연산을 효율적으로 진행하기 위해 DF 내부에 왼쪽 참조 픽셀을 저장하기 때문이다. 그러나 위쪽 참조 매크로 블록도 DF 내부에 저장을 한다면 영상 사이즈가 커질수록 너무 많은 저장 공간이 필요하게 되므로 DF 외부의 공유 SRAM에 저장한 다음, 필요할 때 SRAM에서 원하는 참조 매크로 블록을 불러와서 연산을 한다.

픽셀 재배치 유닛은 수직경계의 필터연산을 마친 픽셀을 반대방향으로 재배치함으로써 수평경계의 필터연산을 할 때 효율적으로 필터연산을 할 수 있도록 한다. 수직경계의 필터연산은 1 사이클에 필요한 픽셀을 불러와서 연산이 가능하지만, 수직경계의 필터연산 후 픽셀의 재배치를 하지 않는다면, 수평경계의 필터연산은 필요한 픽셀 데이터를 불러오는데 4 사이클이 소모되어 비효율적으로 된다. 이러한 구조를 개선시키기 위한 픽셀을 재배치하는 유닛을 사용했으며, 그 구조가 그림 6에 나타나 있다. 재배치 유닛은 8 bit 레지스터 16 개와 32 bit 레지스터 4 개를 사용해서 구현하였다. Port에는

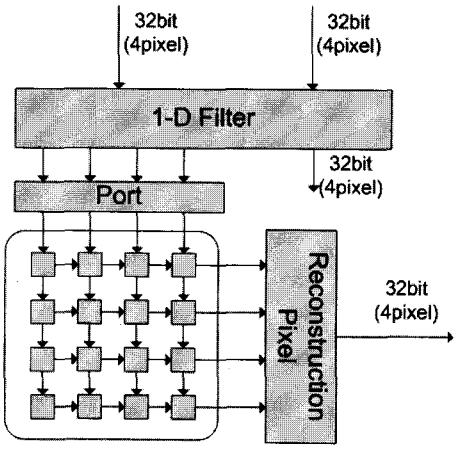


그림 6. 픽셀 재배치 유닛의 구조
Fig. 6. Structure of reconstruction unit.

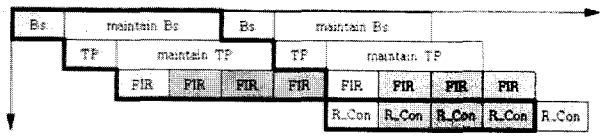


그림 7. DF 연산 파이프라인 구조
Fig. 7. Pipelined architecture of the proposed deblocking filter.

4 x 4 픽셀을 저장하고 그 저장된 4 x 4 픽셀을 16 개의 8 bit 레지스터에 저장한다. 다음 4 사이클 동안 각 8 bit 레지스터에 들어간 픽셀 값을 오른쪽으로 쉬프트해서 4 픽셀씩 메모리에 저장한다. 이 4 사이클 동안 Port에서는 1-D 필터에서 나오는 4 x 4 출력 픽셀 값을 저장한다.

제안된 DF 구조는 크게 Bs 계수 연산, ROM 테이블에서 α, β, t_{C0}의 임계값 읽기, 필터연산의 3 단계로 나누었다. DF 연산의 3 단계를 1 사이클에 수행할 경우 높은 주파수에서 동작이 불가능하고, 영상 사이즈가 커짐에 따라 더 높은 동작 주파수가 필요하므로 파이프라인 구조로 설계하였다. 그림 7은 본 논문이 제안한 파이프라인 구조를 사이클 단위로 나타낸 것이다. 그림 7의 R_con은 재배치된 픽셀을 의미하며 TP는 α, β, t_{C0}를 의미한다.

III. 설계 및 구현

제안된 구조에 따라 DF를 Verilog-HDL을 이용하여 설계하였다. AMBA AHB 버스에 연결하기 위해 AHB 인터페이스를 가지고 있고 데이터 전송을 요청하기 위한 인터럽트 요청 기능이 포함되었다. 설계된 DF의 동작 사이클 수를 살펴보면 메모리에서 픽셀을 읽고 저장할 때 지연되는 사이클이 없다고 가정한다면 DF는 휘도 픽셀을 연산 할 경우 151 사이클이 필요하다. 그중에 128 사이클은 필터연산에 필요하고 나머지 19 사이클은 파이프라인 지연과 최종 픽셀 값을 저장하는데 사용된다. 색차 데이터 연산의 경우 79 사이클을 사용하며 64 사이클은 필터연산에 쓰이고, 나머지 15 사이클은 휘도와 동일하다. 따라서 하나의 매크로 블록에 대한 DF의 총 연산은 230 사이클을 필요로 한다.

설계된 결과물을 Synopsys Design Complier와 0.28um 표준 CMOS 공정을 이용하여 합성하였다. 표1에 합성 및 비교자료가 나타나 있다. DF 내부 메모리 크기와 메모리 종류, 합성 결과의 게이트 수, 매크로블록 당 소모되는 사이클을 비교하였다. DF가 매크로블록을 저장하는데 사용하는 메모리 크기는 (64 + 32 + 32 + 16) x 32 = 4,608 bit이다. 본 논문에서 제안하는 DF의 효율적인 메모리 관리를 통해 기존 결과에 비해 메모리 크기를 10% 정도 감소시키면서 동작 사이클 수도 줄일 수 있었고 1-D 필터를 사용하여 DF의 로직 면적도 상당히 감소시킬 수 있었다. 설계된 구조는 AHB 인터페이스를 제외한 면적이다. 제안된 DF의 동작 주파수인 77 MHz는 DF가 HD급 영상을 처리할 경우 필터링 데이터를 저장하는 시간까지 포함한 것으로 필터링 연산만을 고려한다면 더 낮은 주파수에서도 동작 가능하다. 합성 결과 최대 동작 주파수는 200 MHz까지 가능하였다. [3]의 경우 괄호안의 값은 HD 해상도에서 동작하는데 필요한 최소 동작 주파수이다.

디블록킹 필터가 별도의 지연 사이클 없이 참조 프레

표 1. Deblocking 필터의 합성 결과 및 비교
Table 1. Comparison results of synthesized deblocking filters.

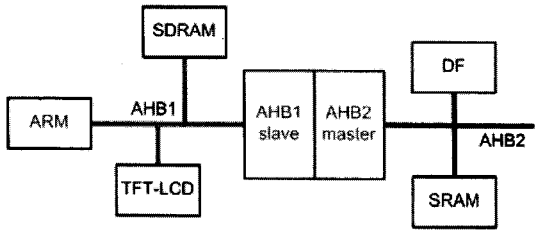
	[1]	[2]	[3]	Proposed
Memory [bit]	5,120	5,120	5,120	4,608
Memory type	dual port	dual port	single port	dual port
Gate count	9,350	20,660	19,640	11,250*
cycle/MB**	566	614	250	230
Operating Frequency [MHz]	100	100	100 (85**) (HD)	77 (HD)***
Technology	0.35um	0.25um	0.18um	0.25um

* AHB 인터페이스 제외
** 순수한 필터링 연산에 필요한 사이클 수
***HD : 1920 x 1080, 30fps

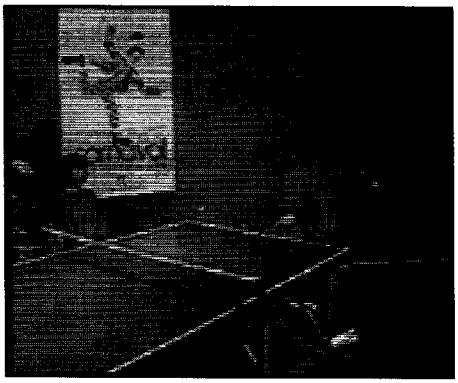
표 2. 다양한 영상 크기에서 필요한 동작 주파수
Table 2. Operating frequency for decoding various image sizes.

	QCIF*	QVGA**	D1***	HD
Minimum Operating Frequency [MHz]	1	3	13	77
Number of MB	99	300	1,350	8,160

*QCIF : 176 x 144, 30fps
**QVGA : 320 x 240, 30fps
***D1 : 720 x 480, 30fps



(a)



(b)

그림 8. DF 블록의 검증 시스템 및 검증 결과 (a) 검증 시스템의 블록 다이어그램 (b) 설계된 DF를 통과한 영상
Fig. 8. Verification of the implemented deblocking filter (a) Block diagram of the platform for verification (b) Filtered image displayed on the verification system.

임 메모리에 픽셀값을 저장한다면, 최소 96 사이클을 소모한다. 따라서 DF의 연산 사이클과 참조 프레임에 저장되는 사이클은 매크로 블록 당 326 사이클을 소모한다. DF의 연산 중에 다음 연산에 필요한 픽셀이 DF에 모두 저장된다면, HD(1920 x 1080, 8,160 MB/frame) 해상도의 4:2:0포맷의 영상을 복호화하기 위한 필요 동작 주파수는 77 MHz가 된다. 표 2에 영상의 해상도별로 설계 DF를 이용하여 복호화하기 위해 필요한 최소 동작 주파수가 나타나 있다. 표 1에서 [1]과 [2]는 HD 해상도에서는 동작이 안 되고 [3]은 85MHz에 가능하고 면적은 거의 2배이다.

설계된 DF 블록의 검증은 FPGA에서 진행하였고 그림 8에 검증 시스템과 검증 결과가 나타나 있다. DF 블록이 AHB 인터페이스를 가지고 있으므로 그림 8(a)에 나타난 바와 같이 ARM 기반 검증 플랫폼과 JM9.0 소프트웨어를 연동하여 검증하였다. 그림 8(b)에서 필터를 통과한 영상이 정상적으로 출력됨을 확인할 수 있다.

Ⅳ. 결 론

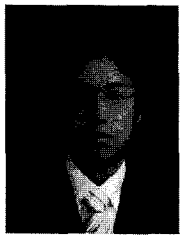
본 논문에서는 효율적인 메모리 관리와 파이프라인 구조를 갖는 H.264용 디블록킹 필터 구조를 제안하였다. DF와 MC간의 병목현상 제거를 위하여 휘도성분과 색차성분 2개를 이중 버퍼링 메모리 구조를 이용하여 순차적으로 저장할 수 있도록 하여 DF와 MC 사이 또는 MC에 별도의 메모리를 두지 않고 병목현상 제거가 가능하였고 H.264 디코더의 성능 향상의 이점을 얻었다. 또한, 제안된 연산 순서는 파이프라인 지연으로 발생되는 연산 사이클을 제거하는 이점을 얻었으며 효율적인 메모리 관리를 위하여 참조 블록을 저장하는 메모리를 공유함으로써 전체적인 메모리 크기를 감소시켰다. 한편, 수직, 수평경계의 필터연산을 할 수 있는 1-D 필터를 재사용하여 필터의 수를 줄이고 필터 면적의 최소화 및 파이프라인 구조가 가능하도록 하였다. 제안된 구조에 따라 설계된 DF를 기존의 결과와 비교한 결과 메모리 크기를 10% 정도 줄였고 로직 면적도 감소시켰다. 그 결과 77 MHz에서 HD급 영상에 대해서도 필터링이 가능함을 보였다.

참 고 문 헌

[1] Lingfeng Li, Goto S, Ikenage T, "An efficient deblocking filter architecture with 2-dimensional parallel memory for H.264/AVC", Proceedings of the ASP-CAC 2005, Vol. 1, pp. 623-626, 18-21 Jan 2005.
[2] Yu-Wen Huang, To-Wei Chen, Bing-Yu Hsieh, Tu-Chih Wang, Te-Hao Chang, and Liang-Gee Chen "ARCHITECTURE DESIGN FOR DEBLOCKING FILTER IN H.264/JVT/AVC," ICME '03 Proceeding, vol.1, pp. 693-696, 6-9 July 2003.
[3] T. M. Liu, W. P. Lee, T. A. Lin, C. Y. Lee

"A memory-efficient deblocking filter for H.264/AVC video coding" in Proc. IEEE ISCAS, pp. 2140-2143, 23-26 May 2005.
[4] *Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC)*, ITU-T JVT G050, 2003.
[5] Iain E.G. Richardson, H.264 and MPEG-4 VIDEO COMPRESSION, John Wiley & Sons, pp.159-224, Jan 2003.
[6] Miao Sima, Yuanhua Zhou, and Wei Zhang, "An Efficient Architecture for Adaptive Deblocking Filter of H.264/AVC Video Coding" IEEE Transactions on Consumer Electronics, Vol. 50, No. 1, pp. 292-296, Feb 2004.
[7] H.264/AVC Reference Software JM 9.0, ITU-T, JVT, Nov. 2005.

저 자 소 개



유 용 훈(학생회원)
2007년 숭실대학교 정보통신전자공학부 학사졸업.
2007년~현재 숭실대학교 전자공학과 석사재학.
<주관심분야 : H.264 코덱 구현, SoC 설계방법론, SoC 플랫폼 설계>

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