논문 2009-46SD-2-13

하이브리드 구조를 갖는 MPEG-4 인코더용 전역 탐색 블록 정합 움직임 추정 회로

(Full-Search Block-Matching Motion Estimation Circuit with Hybrid Architecture for MPEG-4 Encoder)

심 재 오*, 이 선 영*, 조 경 순**

(Jaeoh Shim, Seonyoung Lee, and Kyeongsoon Cho)

요 약

본 논문은 시스톨릭 어레이와 덧셈기 트리를 조합한 하이브리드 구조를 갖는 MPEG-4 인코더용 전역 탐색 블록 정합 움직임 추정 회로를 제안한다. 제안된 회로는 적은 수의 클럭 싸이클로 움직임 추정을 할 수 있도록 시스톨릭 어레이를 활용하고, 필요한 회로 자원을 줄이기 위해서 덧셈기 트리를 활용한다. 1/2화소 움직임 추정을 위한 보간 회로는 6개의 덧셈기, 4개의 뺄셈기, 10개의 레지스터로 구성하였으며, 자원 공유 및 효율적인 스케줄링 기법을 통하여 성능을 향상시켰다. 정수화소 및 1/2 화소를 위한 움직임 추정 회로를 Verilog HDL을 사용하여 RTL에서 설계하였다. 130nm 표준 셀 라이브러리를 사용하여 합성한 논리 수준 회로는 218,257 게이트로 구성되었으며, D1(720x480) 이미지를 초당 94장 처리할 수 있다.

Abstract

This paper proposes a full-search block-matching motion estimation circuit with hybrid architecture combining systolic arrays and adder trees for an MPEG-4 encoder. The proposed circuit uses systolic arrays for motion estimation with a small number of clock cycles and adder trees to reduce required circuit resources. The interpolation circuit for 1/2 pixel motion estimation consists of six adders, four subtracters and ten registers. We improved the circuit performance by resource sharing and efficient scheduling techniques. We described the motion estimation circuit for integer and 1/2 pixels at RTL in Verilog HDL. The logic-level circuit synthesized by using 130nm standard cell library contains 218,257 gates and can process 94 D1(720x480) image frames per second.

Keywords: MPEG-4 인코더, 움직임 추정, 전역 탐색 블록 정합 알고리즘, 보간

I. 서 론

MPEG-4 Part2: Visual[1]에서 움직임 추정은 상당히 많은 양의 연산을 필요로 하기 때문에 실시간 비디오 압축을 할 때 가장 중요한 요소이며, 많은 알고리즘이 연구되었다[2]. 전역 탐색 블록 정합 알고리즘, 즉 FSBMA(Full-Search Block-Matching Algorithm)는 높은 정확성과 규칙성으로 인해 움직임 추정 회로를 설계할 때 가장 널리 사용된다. FSBMA를 회로로 구현하기 위한 다양한 구조가 연구되었는데, 대표적으로 시스톨릭 어레이 구조[3~6]와 트리 구조[7~9]가 있다. 시스톨릭 어레이 구조를 사용하는 회로는 상대적으로 적은 클럭 싸이클 수를 필요로 하며, 트리 구조를 사용하는 회로는 자원을 적게 사용한다. 본 논문은 이 두 구조의 장점을 취한 하이브리드 구조 및 1/2화소 단위의 움직임 추정을 위한 보간 회로[10~11]의 구조를 제안한다. 효율적인 보간 회로를 설계하기 위하여 연산기와 레지스터를 공유함으로써 회로의 크기를 줄였고, 적절한 스케줄링 기법을 통하여 클럭 싸이클 수를 감소시켰다. 본 논

* 학생회원, ** 평생회원, 한국외국어대학교
전자정보공학부
(Department of Electronics and Information Engineering, Hankuk University of Foreign Studies)
※ 이 연구는 2008학년도 한국외국어대학교 교내학술연구비의 지원에 의하여 이루어진 것임.
접수일자: 2008년12월31일, 수정완료일: 2009년2월2일

문은 4개의 절로 구성되어 있다. Ⅱ장에서는 FSBMA, 제안한 구조, 회로 타이밍 분석에 대해 설명한다. Ⅲ장에서는 실험 결과를 기술하고 있으며, Ⅳ장에서는 결론을 제시한다.

Ⅱ. 회 로 구 조

1. Full-Search Block-Matching Algorithm

그림 1에서 볼 수 있듯이, MPEG-4 인코더의 움직임 추정 회로는 현재 프레임 F_n의 매크로블록 픽셀 데이터와 참조 프레임 F'_{n-1}의 탐색 영역 픽셀 데이터를 입력으로 받는다. FSBMA에서 현재 프레임은 NxN 블록 크기의 매크로블록으로 구성되어 있다. 그림 2와 같이 'Current frame'의 각 매크로블록에 대해 'Reference frame'이라 부르는 이전 프레임의 일정 탐색 영역에서 가장 유사한 블록을 탐색하게 된다. 여기서 p는 탐색 범위이다. 그림 3은 N=3, p=2인 경우의 매크로블록과 탐색 영역을 보여준다. m(i,j)는 매크로블록의 각 픽셀을 의미하며, s(i,j)는 탐색 영역의 각 픽셀을 의미한다. 탐색 영역은 총 (N+2p)x(N+2p)개의 픽셀로 구성된다.

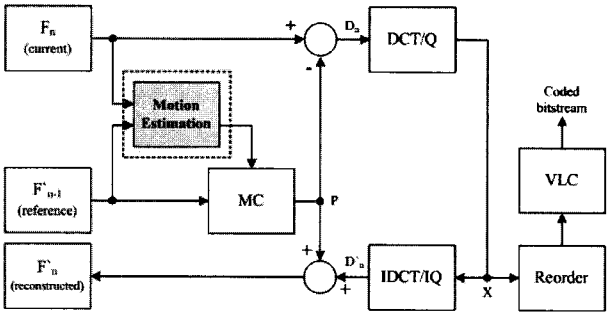

그림 1. MPEG-4 인코더의 블록도
Fig. 1. Block diagram of MPEG-4 encoder.

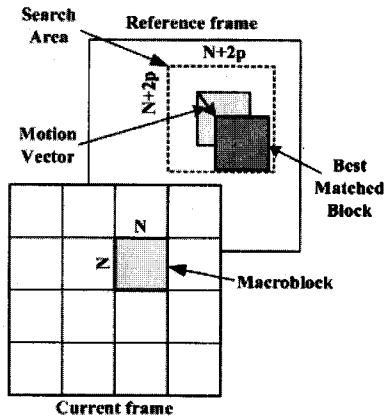

그림 2. FSBMA를 사용한 움직임 추정 과정
Fig. 2. Motion estimation process using FSBMA.

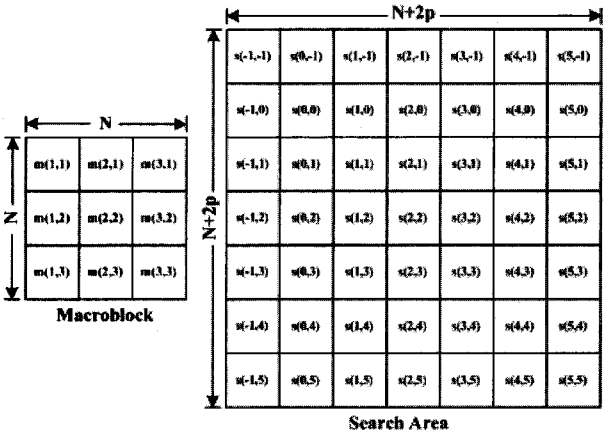

그림 3. N=3, p=2 경우의 매크로블록과 탐색 영역
Fig. 3. Macroblock and search area for N=3, p=2.

SAD(Sum of Absolute Difference)는 두 매크로블록 간의 유사성을 나타내는 척도이며, 식 (1)과 같이 구한다.

$$SAD(x,y) = \sum_{i=1}^{N}\sum_{j=1}^{N} |m(i,j) - s(i+x, j+y)| \quad for\ -p \le \{x,y\} \le p \tag{1}$$

2. 제안한 구조

(1) 하이브리드 구조

FSBMA를 위한 회로 구조 중 가장 널리 쓰이는 구조는 시스톨릭 어레이와 덧셈기 트리이다. 시스톨릭 어레이 구조가 데이터 재사용 기법을 이용하여 클럭 싸이클 수를 줄일 수 있는 반면, 트리 구조는 회로의 자원을 적게 쓸 수 있다. 즉, 한쪽 구조의 장점이 다른 한쪽 구조의 단점이 된다. 그림 4와 5는 각각 두 구조를 보여준다. 본 논문에서는 두 구조의 장점을 모두 가진 하이브

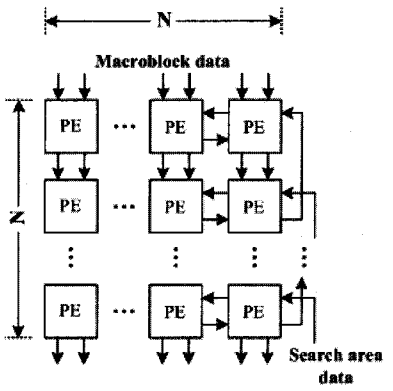

그림 4. 시스톨릭 어레이 구조
Fig. 4. Systolic array structure.

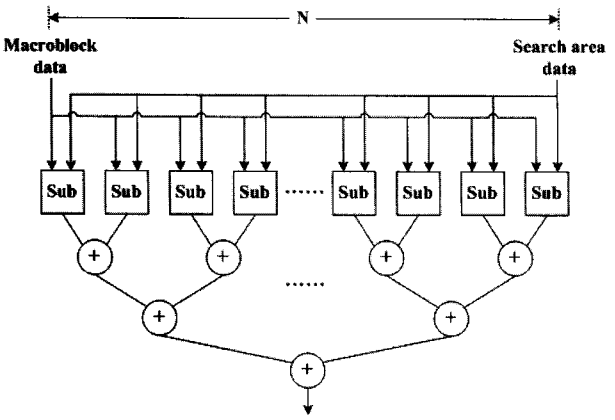

그림 5. 덧셈기 트리 구조
Fig. 5. Adder tree structure.

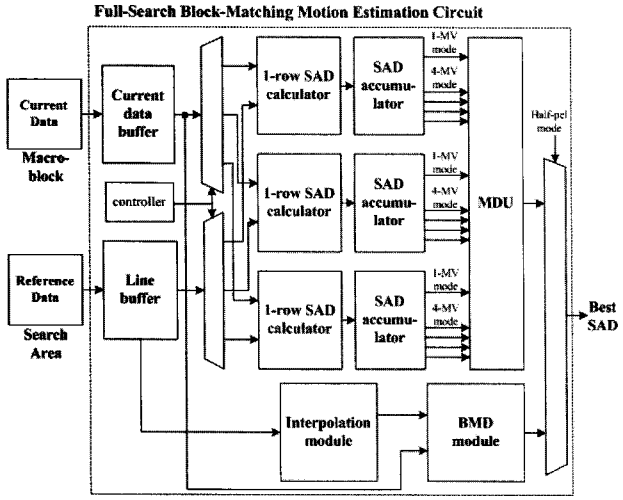

그림 6. N=3, p=2 경우의 제안한 움직임 추정 회로 구조
Fig. 6. Proposed architecture of motion estimation circuit for N=3, p=2.

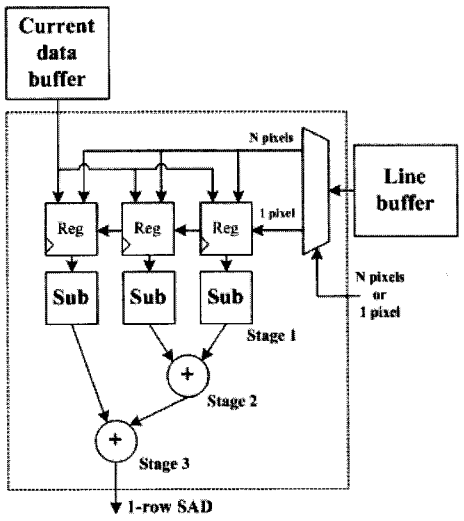

그림 7. N=3, p=2 경우의 '1-row SAD calculator' 블록
Fig. 7. '1-row SAD calculator' block for N=3, p=2.

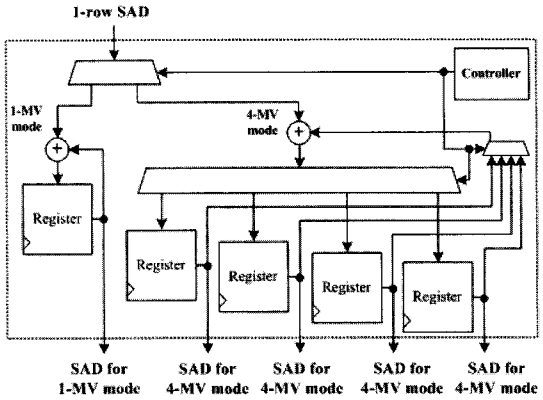

그림 8. N=3, p=2 경우의 'SAD accumulator' 블록
Fig. 8. 'SAD accumulator' block for N=3, p=2.

리드 구조를 제안한다. 그림 6은 N=3, p=2인 경우의 움직임 추정 회로를 보여준다. '1-row SAD calculator' 블록은 현 매크로블록의 픽셀 데이터와 참조 프레임의 탐색 영역의 픽셀 데이터에 대한 뺄셈 연산을 수행한다. 'SAD accumulator' 블록은 현재의 매크로블록에 대한 SAD를 저장한 후, 가장 작은 SAD를 결정하기 위해 'MDU(Minimum Decision Unit)' 블록으로 전송한다.

'Interpolation module' 블록과 'BMD(Best Matching-block Decision) module' 블록은 1/2화소 단위의 움직임 추정에 사용된다. 'Current data buffer' 블록은 NxN 크기의 매크로블록을 저장하기 위한 것이고, 'Line buffer' 블록은 참조 프레임으로부터 받은 N+1 픽셀 데이터를 저장하여 탐색 영역 데이터로 사용하기 위한 것이다. 'controller' 블록은 버퍼 블록들에서 '1-row SAD calculator' 블록들로 데이터를 전송하는 타이밍을 조절해준다.

'1-row SAD calculator' 블록은 'Current data buffer' 블록과 'Line buffer' 블록으로부터 입력을 받는다. 그림 7은 N=3, p=2인 경우의 블록도를 보여준다. 'Line buffer' 블록은 첫 번째는 N개의 픽셀을 공급하지만, 두 번째부터는 1 픽셀씩 공급하게 된다. 결과적으로 N-1개의 픽셀을 재사용하여 회로의 동작 속도를 높이게 된다. N=3인 경우, 덧셈기 트리를 사용하여 매크로블록 데이터와 탐색 영역의 데이터의 차분 값을 하나의 '1-row SAD' 값으로 만들기 위해서는 두 클럭 싸이클이 소모된다. 이를 파이프라인 기법을 이용하여, 한 클럭 싸이클만에 매크로블록의 한 줄인 N개의 픽셀에 대한 '1-row SAD'값을 얻도록 하여 동작 성능을 높였다.

그림 8은 'SAD accumulator' 블록을 보여준다. '1-row SAD calculator' 블록에서 출력된 '1-row SAD'

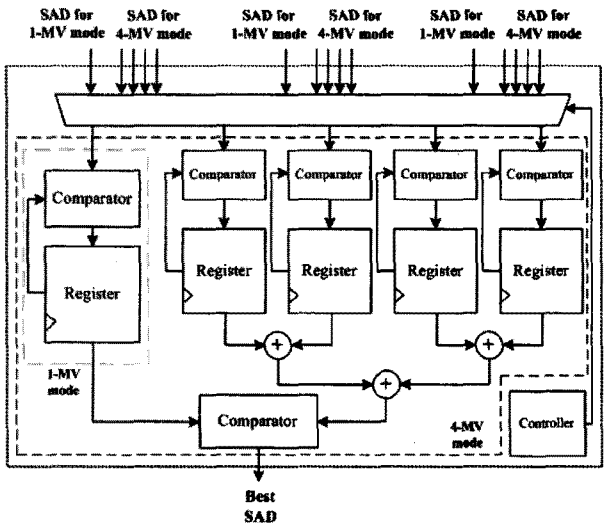

그림 9. N=3, p=2 경우의 'Minimum Decision Unit' 블록
Fig. 9. 'Minimum Decision Unit' block for N=3, p=2.

를 입력을 받아 한 매크로블록에 해당하는 SAD가 되도록 축적한다. 'Controller' 블록은 입력되는 '1-row SAD'가 움직임 벡터가 1개인 '1-MV mode'인지, 4개인 '4-MV mode'인지 구분하여 알맞은 레지스터에 담길 수 있도록 한다.

'SAD accumulator' 블록의 레지스터들은 탐색 영역을 나타내는 p에 따라서 개수가 결정된다. 즉, '1-MV mode'를 위한 레지스터 개수는 2p개이고 '4-MV mode'를 위한 레지스터 개수는 8p개가 된다. 축적되는 SAD가 한 개의 매크로블록에 대한 값이 되면 해당하는 모드에 따라 'SAD for 1-MV mode' 및 'SAD for 4-MV mode'를 출력하여 최종 블록인 'MDU' 블록으로 전달하게 된다.

그림 9에서 볼 수 있듯이, 'MDU' 블록은 'SAD accumulator' 블록에서 출력된 SAD를 '1-MV mode'와 '4-MV mode'에 따라 구분된 입력으로 받는다. 만약 '1-MV mode'일 경우에는 'MDU' 블록은 'SAD for 1-MV mode'만 입력으로 받은 후, 그것을 해당하는 레지스터에 저장한다. 그 후에 연속해서 들어오는 SAD를 기존의 저장되어 있던 SAD와 비교하여 만약 더 작다면 교체하여 저장하고, 그렇지 않다면 기존의 저장되어 있던 것이 남게 되는 동작을 반복하여 최종적으로 'Best SAD'로 출력한다. '4-MV mode'일 경우에는 5개의 레지스터와 비교기('Comparator')가 위와 같은 동작을 하여 최종적으로 '1-MV mode'의 최소 SAD와 '4-MV mode'의 4개의 최소 SAD의 합을 비교하여 최종적으로 가장 작은 SAD인 'Best SAD'를 출력한다.

(2) 보간 회로 구조

본 논문에서 제시한 보간 회로는 2개의 모듈로 구성된다. 하나는 'Interpolation module' 블록이고, 다른 하나는 'BMD module' 블록이다. 'Interpolation module' 블록은 정수화소로부터 1/2화소를 얻어내기 위한 모듈이고, 연산된 1/2화소로부터 최소 SAD 값을 갖는 부분을 찾는 모듈이 'BMD 모듈' 블록이다. 그림 10은 보간 방법을 보여준다. 'aa' 와 'bb'의 1/2화소들은 'A', 'B', 'C'의 정수 단위 픽셀의 평균값으로 얻는다. 'cc'의 경우에는 둘러싼 네 개의 정수픽셀의 평균값으로부터 얻는다.

그림 11은 'Interpolation module' 블록을 보여준다. 정수화소 4개로부터 1/2화소 3개를 얻어내는 과정에서 첫 번째 연산 시, 'B'와 'D'에 대한 덧셈 연산의 결과가 다음 연산 시 'A'와 'C'의 덧셈 연산과 겹친다. 이 부분을 이용하여 덧셈기 2개, 레지스터 2개를 공유하여 회로를 설계하였다. 'Controller' 블록은 덧셈기와 레지스터가 사용되는 타이밍을 스케줄링한다.

'Interpolation module' 블록으로부터 출력된 화소들은 'BMD module'로 입력된다. 1개의 정수화소는 그림 12

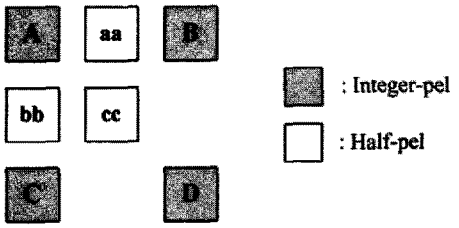

그림 10. 보간 방법
Fig. 10. Interpolation method.

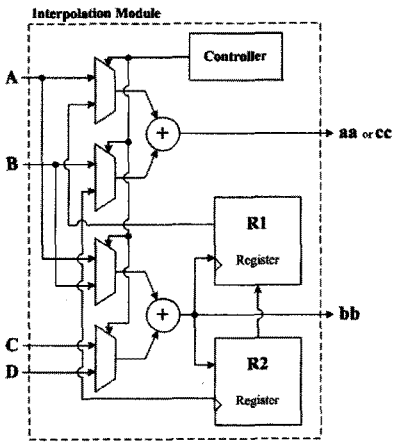

그림 11. 'Interpolation module' 블록
Fig. 11. 'Interpolation module' block.

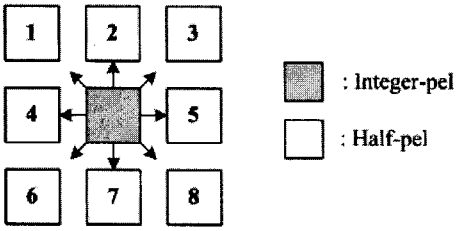

그림 12. 한 개의 정수화소에 대한 1/2화소의 위치
Fig. 12. Location of half-pels for an integer-pel.

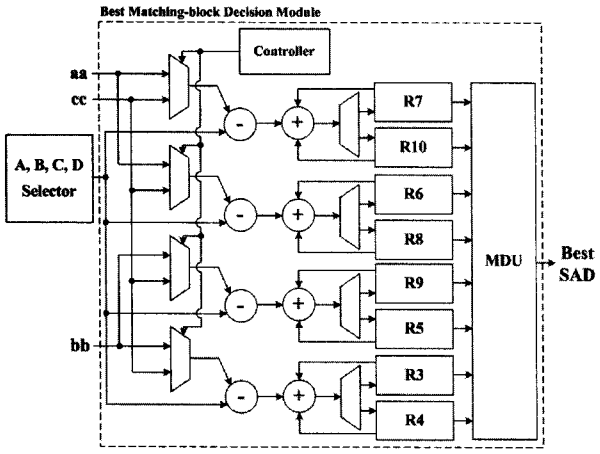

그림 13. 'BMD module' 블록
Fig. 13. 'BMD module' block.

와 같이 8방향의 1/2화소와 비교하게 된다. 각 8개 방향의 1/2화소와 비교하기 위해서 레지스터 8개를 사용하였고, 그 구조는 그림 13에서 보여주는 것과 같다. 'A, B, C, D Selector'는 1/2화소의 각 비교 방향이 맞도록 정수화소를 선택해주어 4개의 뺄셈기로 입력하는 역할을 한다. 'Controller' 블록은 뺄셈기 4개로 입력되는 1/2화소들을 적절하게 배분하기 위한 것이다. 덧셈기 4개는 'R3'부터 'R10'까지의 8개 레지스터로 1/2화소 단위의 SAD를 축적하기 위한 것이다. 한 개의 매크로블록에 대해 1/2화소 단위의 SAD 연산이 끝나면 'MDU(Minimum Decision Unit)' 블록에 전달되어 가장 작은 값의 SAD를 'Best SAD'로 출력한다.

3. 회로 타이밍 분석

(1) 정수화소 단위의 움직임 추정 회로 타이밍 분석

'1-row SAD calculator' 블록에서 사용하는 파이프라인의 동작은 그림 14와 같다. '1', '2', '3'은 파이프라인의 각 단계를 뜻한다. '1'은 'Current data buffer' 블록으로부터 받은 픽셀들과 'Line buffer' 블록으로부터 받은 픽셀들 간의 뺄셈 연산을 뜻한다. '2'는 뺄셈 연산된 픽셀

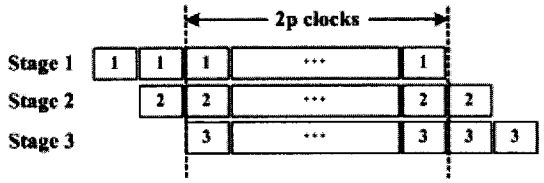

그림 14. '1-row SAD calculator' 블록의 파이프라인
Fig. 14. Pipeline of '1-row SAD calculator' block.

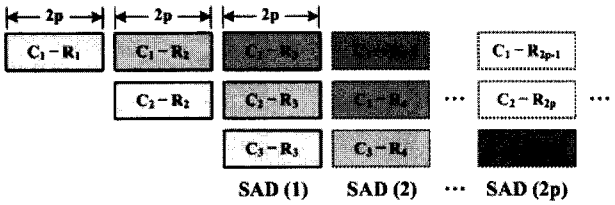

그림 15. '1-row SAD calculator' 블록의 병렬 구조
Fig. 15. Parallel structure of '1-row SAD calculator' block.

들을 덧셈기를 사용하여 더해주는 연산이며, '3' 역시 덧셈기를 사용하여 '1-row SAD'를 얻는 과정을 뜻한다. 결과적으로 N개의 픽셀에 대한 SAD('1-row SAD')를 얻기 위해서는 한 클럭 싸이클이 필요하기 때문에, 2p개의 픽셀에 대한 SAD를 얻기 위해서는 2p 클럭 싸이클을 필요로 한다.

본 논문에서 제안한 회로는 데이터 재사용, 파이프라인, 병렬 구조와 같은 기법들을 사용하여 회로 동작에 필요한 싸이클 수를 감소시켰다. N개의 '1-row SAD calculator' 블록은 N개 열의 매크로블록 개수만큼 병렬 구조를 가진다. 그 동작을 살펴보면 다음과 같다.

첫째, '1-row SAD calculator' 중 첫 번째 블록은 'Current data buffer' 블록으로부터 전송된 현재 매크로블록의 첫 번째 줄과 'Line buffer'로부터 전송된 탐색 영역의 첫 번째 줄을 연산하여 SAD를 얻는다. 두 번째와 세 번째 '1-row SAD calculator' 블록들은 첫 번째 '1-row SAD calculator' 블록이 연산이 끝날 때까지 기다리게 된다. 그림 15는 그 과정을 보여준다. 'C_a'와 'R_b'는 각각 'Current data buffer'로부터 나온 매크로블록의 a번째 줄과 'Line buffer'로부터 나온 탐색 영역의 b번째 줄을 의미한다.

둘째, 첫 번째 '1-row SAD calculator' 블록은 현재 매크로블록의 두 번째 줄과 탐색 영역의 두 번째 줄을 연산하여 SAD를 얻는다. 따라서 첫 번째 단계에 이어 매크로블록의 위치가 탐색 영역의 첫 번째 줄에 위치할 때의 SAD를 얻기 위해서는 한 번의 계산이 더 필요한 셈이다. 두 번째 '1-row SAD calculator' 블록은 현재

매크로블록의 첫 번째 줄과 탐색 영역의 두 번째 줄을 가지고 SAD를 연산한다.

셋째, 첫 번째 '1-row SAD calculator' 블록은 매크로블록의 세 번째 줄과 탐색 영역의 세 번째 줄을 연산하여 SAD를 얻는다. 두 번째 '1-row SAD calculator' 블록은 매크로블록의 두 번째 줄과 탐색 영역의 세 번째 줄을 연산하여 SAD를 얻는다. 이는 매크로블록이 탐색 영역의 두 번째 줄에 위치할 때의 SAD를 얻기 위해선 한 단계가 더 필요하다는 것을 의미한다. 세 번째 '1-row SAD calculator' 블록은 매크로블록의 첫 번째 줄과 탐색 영역의 세 번째 줄을 연산하여 SAD를 얻게 되는데, 이는 매크로블록이 탐색 영역의 세 번째 줄에 위치할 때의 SAD 연산을 시작하게 됨을 의미한다.

첫째부터 셋째까지의 과정을 반복하여 탐색 영역의 모든 후보 블록들과 매크로블록 간의 비교가 이루어지게 되며, 소모되는 총 클럭 싸이클 수는 2px2p이다. 첫 번째 '1-row SAD calculator' 블록은 탐색 영역의 첫 번째 줄에 위치한 매크로블록에 대해 SAD 연산이 모두 끝나면, 다시 매크로블록이 탐색 영역의 네 번째 줄에 위치할 때의 SAD를 구하기 위한 연산을 시작한다.

(2) 보간 회로 타이밍 분석

그림 10에서 볼 수 있듯이, 1/2화소 단위의 픽셀을 얻기 위한 식은 3가지 종류가 있다. 하지만 그 세 가지의 식을 정리하면 1/2화소를 얻기 위한 클럭 싸이클 소모량을 크게 감소시킬 수 있다. 그림 16은 '인터폴레이션 모듈' 블록과 'BMD 모듈' 블록을 통해 동작하는 보간 회로의 동작 타이밍을 보여준다. 4개의 정수화소로부터 3개의 1/2화소를 얻어내는데 두 클럭 싸이클이 소요된다.

블록의 처음 시작하는 부분에서는 동작의 준비를 위한 'Initial clock'이 필요하다. 또한 효율적인 스케줄링을 통하여, 'aa', 'bb', 그리고 'cc'가 출력되는 동시에 'BMD 모듈'로 입력되어 바로 SAD 비교를 위한 동작이 연결된다. 결과적으로 한 개의 매크로블록을 처리하는데 소요되는 클럭 싸이클 수는 식 (2)와 같다. 'N'은 매크로블록 한 변의 픽셀 수를 의미한다.

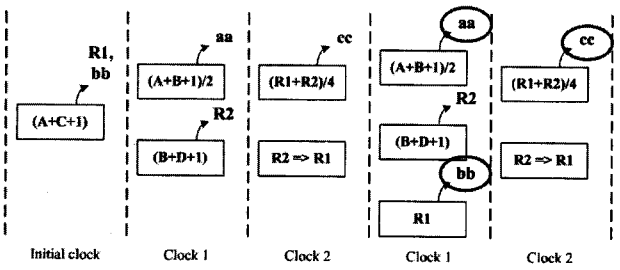

그림 16. 보간 회로의 동작 타이밍도
Fig. 16. Timing diagram of interpolation circuit.

$$Total\ Clock\ Cycles = [1 + \{(N+1)\times 2\}]\times(N+1) \quad (2)$$

Ⅲ. 실 험 결 과

본 논문은 하이브리드 구조를 갖는 MPEG-4 인코더용 전역 탐색 블록 정합 움직임 추정 회로를 제안한다. 제안된 회로는 Verilog HDL(Hardware Description Language)를 사용하여 RTL(Register Transfer Level)로 구현하였으며, Cadence사의 NC-Verilog를 이용하여 검증하였다. Synopsys사의 Design Compiler를 통해 130nm 표준 셀 라이브러리를 이용하여 논리 수준 회로로 합성한 결과, 218,257 게이트로 구성되었으며 최대 동작주파수는 116MHz이다. 합성된 논리 수준 회로는 D1(720x480) 영상을 초당 94장 처리할 수 있다. 데이터 재사용, 파이프라인, 병렬 구조, 스케줄링, 그리고 연산기 및 레지스터 공유와 같은 기법을 이용하여 설계한 정수화소 및 1/2화소를 위한 움직임 추정 회로는 기존의 회로들보다 회로 동작에 필요한 클럭 싸이클 수가 적을 뿐만 아니라 회로 크기도 작다.

표 1은 본 논문에서 제안한 회로의 실험 결과를 보여준다. 여기서 '전체 회로'는 연결 회로를 포함한 전체 회로를 의미한다. 표 2는 FSBMA를 사용한 회로들이 동작에 필요로 하는 클럭 싸이클 수를 비교한 것이고, 표 3은 N=16, p=15인 경우의 필요 자원을 비교한 것이다. 'Proposed'는 본 논문에서 제안한 회로를 의미한다. 표 2와 3은 정수화소 단위의 움직임 추정 회로에 대한 자

표 1. 제안한 회로에 대한 구현 결과
Table 1. Implementation results of proposed circuit.

	전역 탐색 움직임 추정 회로	보간 회로	전체 회로
Clock Cycles/1MB (N=16, p=15)	916	595	916
동작주파수 (MHz)	116	191	116
합성된 게이트 수	192,772	16,826	218,257

표 2. 동작에 필요한 클럭 싸이클 수 비교
Table 2. Comparison of required number of clock cycles.

	처리 시간	(N, p)		
		(3,2)	(8,7)	(16,15)
[4]	$(2p+1)(N+2p)$	35	330	1,426
[5]	$N^2+(N+2p)(2p+1)+1$	47	402	1,698
[6]	$(N+2p)(p+3)$	35	220	828
[7]	$16\{(2p+1)^2+2\log_2 N+1\}$	465	3,712	15,520
[8]	$(2p+1)^2+2\log_2 N+1$	29	232	970
Proposed	$N+(2p)^2$	19	204	916

표 3. N=16, p=15 경우의 회로 자원 비교
Table 3. Comparison of circuit resources for N=16, p=15.

	덧셈기 수	레지스터 수	
[4]	1,024	8비트	512
		16비트	0
[5]	768	8비트	1,024
		16비트	0
[6]	1,056	8비트	512
		16비트	0
[7]	34	8비트	288
		16비트	16
[8]	512	8비트	992
		16비트	528
Proposed	576	8비트	272
		16비트	496

료이다. [4]~[6]은 시스톨릭 어레이 구조를 사용한 전역 탐색 움직임 추정 회로이고, [7]과 [8]은 트리 구조를 사용한 회로이다. 표 2에서 볼 수 있듯이, N=16이고 p=15일 경우에 회로 동작에 필요한 클럭 싸이클 수는 [6]의 회로가 가장 적다. 하지만 표 3의 회로 자원을 보게 되면, [6]의 회로는 상당히 많은 개수의 덧셈기와 레지스터를 필요로 한다. 반면에 가장 적은 자원을 사용하는 회로는 트리 구조를 사용하는 [7]의 회로이나, 회로 동작에 필요한 클럭 싸이클 수를 보면 아주 작은 이미지만을 실시간으로 처리할 수 있는 수준이다.

전역 탐색 움직임 추정 회로의 대부분의 자원은 덧셈기와 레지스터로 구성되어 있다. 시스톨릭 어레이 구조를 가지는 회로는 덧셈기와 레지스터를 많이 필요로 하지만 빠른 회로 동작 속도를 보인다. 반면에 트리 구조를 가지는 회로는 적은 자원을 사용하여 회로 크기가 작은 대신 상당히 느린 회로 동작 속도를 보여준다. 본

표 4. 보간 회로 비교
Table 4. Comparison of interpolation circuits.

	[10]	[11]	Proposed
Clock Cycles/1MB	980	1,156	595
동작주파수 (MHz)	50	105	191
제조 공정 (nm)	180	250	130
HD 영상에 대한 처리속도 (frames/sec)	6	11	39

논문에서 제안한 회로는 시스톨릭 어레이 구조와 트리 구조의 단점을 보완함과 동시에 장점만을 취하여 빠른 회로의 동작 속도와 적절한 양의 회로 자원으로만 구성되어 MPEG-4 인코더용 고성능 움직임 추정 회로로 사용될 수 있다.

보간 회로를 130nm 표준 셀 라이브러리를 이용하여 논리 수준으로 합성한 결과 16,826 게이트로 구성되었으며, 최대 동작 주파수는 191MHz이다. 이는 HD급 (1,920x1,088) 영상을 초당 39장 처리할 수 있다는 것을 의미한다. 제안한 보간 회로를 연산기와 레지스터의 공유를 이용하여 덧셈기 6개, 뺄셈기 4개, 레지스터 10개로 구성함으로써, 회로 크기를 감소시켰을 뿐만 아니라 효율적인 스케줄링 기법을 통하여 표 4와 같이 다른 보간 회로에 비해 동작에 필요한 클럭 싸이클 수가 적다.

Ⅳ. 결 론

본 논문에서는 MPEG-4 인코더를 위한 하이브리드 구조를 가지는 전역 탐색 블록 정합 움직임 추정 회로를 제안하였다. 전역 탐색 움직임 추정 회로는 기존의 대표적인 구조인 시스톨릭 어레이 구조와 트리 구조의 장점만을 취하여 작은 회로 크기를 갖는 동시에 빠른 회로 동작 속도를 가진다. 보간 회로는 다른 회로처럼 메모리나 버퍼를 사용하지 않고 연산기와 레지스터의 공유를 사용하여 작은 크기의 회로로 구현하였다. 움직임 추정 회로는 탐색 영역에 따라 사용되는 덧셈기와 레지스터의 개수가 결정되며, 전역 탐색임에도 불구하고 넓은 탐색 영역을 실시간으로 처리하는 능력을 보여준다. 또한 방향 벡터가 4개일 경우와 1/2화소 단위의 움직임 추정도 가능하여 고성능 MPEG-4 인코더에 사용되기에 적합하다.

참 고 문 헌

[1] ISO/IEC 14496-2, "Information technology - Coding of audio-visual objects - Part2: Visual," June 2004.

[2] Iain E. G. Richardson, *H.264 and MPEG-4 Video Compression: Video coding for next-generation multimedia*, 2003.

[3] M. A. Elgamel, A. M. Shams and M. A. Bayoumi, "A comparative analysis for low power motion estimation VLSI architectures," *IEEE workshop on Signal Processing Systems 2000*, pp. 149-158, October 2000.

[4] T. Komarck and P. Pirsch, "Array architectures for block matching algorithms," *IEEE Trans. on Circuits and Systems*, vol. CAS-36, pp. 1301-1308, October 1989.

[5] E. Chan and S. Panchanathan, "Motion estimation architecture for video compression," *IEEE Trans. on Consumer Electronics*, vol. 39, issue 3, pp. 292-297, August 1993.

[6] Sungbum Pan, Seungsoo Chae and Raehong Park, "VLSI architecture for block matching algorithm using systolic arrays," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 6, issue 1, pp. 67-73, February 1996.

[7] Y. S. Jehng, L. G. Chen and T. D. Chiueh, "An efficient and simple VLSI tree architecture for motion estimation algorithms," *IEEE Trans. on Signal Processing*, vol. 41, pp. 889-900, February 1993.

[8] Siou-Shen Lin, Po-Chih Tseng and Liang-Gee Chen, "Low-power parallel tree architecture for full search block-matching motion estimation," *Proc. of the 2004 International Symposium on Circuits and Systems*, vol. 2, pp. 313-316, May 2004.

[9] M. Kim, I. Hwang and S. Chae, "A fast VLSI architecture for full-search variable block size motion estimation in MPEG-4 AVC/H.264," *Proc. of the 2005 Asia and South Pacific Design Automation Conference*, vol. 1, pp. 631-634, January 2005.

[10] M. Sayed and W. Badawy, "A half-pel motion estimation architecture for MPEG-4 applications," *Proc. of the 2003 International Symposium on Circuits and Systems*, vol. 2, pp. 792-795, May 2003.

[11] H. Wei-feng, Z. Yan, G. Zhi-qiang and M. Zhi-gang, "Implementation of half-pel motion estimation IP core for MPEG-4 ASP@L5 texture coding," *Proc. of the 2004 Asia-Pacific Conference on Circuits and Systems*, vol. 1, pp. 149-152, December 2004.

저 자 소 개

심 재 오(학생회원)
대한전자공학회논문지
제45권 SD편 제9호 참조

이 선 영(학생회원)
대한전자공학회논문지
제45권 SD편 제9호 참조

조 경 순(평생회원)
대한전자공학회논문지
제45권 SD편 제9호 참조