論文94-31B-12-4

완전탐색에 의한 움직임 추정기 시스토릭 어레이 구조

(Systolic Array Architecture for Full-Search Motion Estimation)

白 鍾 燮*, 南 承 鉉*, 李 文 基*

(Jong Seob Baek, Seung Hyun Nam and Moon Key Lee)

要 約

연속되는 영상의 움직임 보상 부호화에서 가장 자주 사용되는 방법이 블럭정합 움직임 추정기이다. 본 논문에서는 2차원 시스토릭 어레이 구조에 기반을 둔 완전탐색 블럭정합 알고리즘의 집적회로 설계에 효율적인 구조를 제시한다. 제시된 구조는 탐색영역을 효율적으로 구분하여 어레이에 입력하고 이에 알맞은 효율적인 처리기의 설계로 기존의 어레이 구조가 갖는 많은 단점을 없앴다. 그리고 본 구조의 장점은 1) 내부 로칼 메모리의 효율적인 구성을 위해 핀의 개수를 줄이기 위해서 직렬 입력을 하지만 내부적으로 병렬 처리한다. 2) 간단한 제어로 탐색영역의 크기를 임의로 조절할 수 있다. 3) 동작 수행 중에 무효한 연산 사이클이 발생하지 않는다. 4) MPEG-2 의 low 및 high 레벨의 응용에 적합하다. 5) 설계에서 집적회로 화하기 알맞은 모듈성과 규칙성을 가지는 구조이다.

Abstract

Block matching motion estimation is the most widely used method for motion compensated coding of image sequences. Based on a two dimensional systolic array, VLSI architecture and implementation of the full search block matching algorithm are described in this paper. The proposed architecture improves conventional array architecture by designing efficient processing elements that can control the data produced by efficient search window division method. The advantages are that 1) it allows serial input to reduce pin counts for efficient composition of local memories but performs parallel processing. 2) It is flexible and can adjust to dimensional changes of search windows with simple control logic. 3) It has no idle time during the operation. 4) It can operate in real time for low and main level in MPEG-2 standard. 5) It has modular and regular structure and thus is suitable for VLSI implementation.

* 正會員, 延世大學校 電子工學科
(Dept. of Elec. Engi., Yonsei Univ.)
接受日字 : 1994年 3月 22日

I. 서 론

이차원 동화상은 매우 많은 양의 정보를 가지고 있기 때문에 이의 전송을 위해서는 커다란 전송선로와

주파수 대역이 필요하게 된다. 이 문제를 해결하기 위해서는 정보의 압축이 필요하게 되는데 이차원 동화상신호에는 영상정보의 중복성이 존재한다. 이때 시간 축 상의 중복성은 움직임 보상을 통하여 정보 압축할 수 있다. 이때 중요하게 사용되는 것이 이동벡터 추정기이다. 이동벡터 추출에 사용되는 알고리즘으로는 PRA(Pel Recursive Algorithm) 와 블럭정합알고리즘이 있다.[1][2][3][4] 블럭정합알고리즘 중에서 계산 량은 많지만 이동벡터를 상대적으로 정확하게 찾을 수 있는 완전탐색블럭정합 알고리즘이 최근의 집적회로 기술의 발달로 실현 가능해졌다.[5][6] 본 본문은 움직이는 영상의 이동벡터 추출을 위한 알고리즘의 하나인 완전탐색 블럭정합 알고리즘을 이용한 이동벡터 추정기(MEP : Motion Estimation Processor, 이하 MEP라 칭함)의 VLSI구조에 관한 것이다. 블럭정합 알고리즘이란 현재화면을 고정된 크기의 블럭(이하 기준블럭이라 칭함)으로 나누어 각각의 기준블럭이 이전 화면의 정해진 영역에서 블럭단위로 독립적인 변위를 한다는 가정 하에 이루어지는 것으로 이때 이전 화면의 정해진 영역(기준블럭이 -p 에서 +p 화소만큼 상하좌우 이동한 영역,이하 탐색영역이라 칭함)내에 있는 기준블럭의 크기와 같은 블럭단위의 탐색블럭과 기준블럭과의 상관 관계를 구하여 이 값이 최대가 되는 위치 즉 패턴매칭이 가장 잘 이루어지는 위치(이하 이동벡터라 칭함)를 이동벡터로 이용하는 방법이다. 본 논문에서 사용하는 완전탐색 블럭정합 알고리즘을 이용한 이동벡터 추출 일반식은 다음과 같이 나타낸다.

$$ADC_{uv}(i,j) = |S(i+u, j+v) - R(i,k)| \quad (1)$$

$$PS_{uv}(k,j) = \sum_{i=1}^{k} ADC_{uv}(i,j) \quad (2)$$

$$MAD(u,v) = \sum_{j=1}^{N} PS_{u,v}(N,j) \quad (3)$$

수식(1)에서 쓰인 R(i,j)은 기준블럭 내의 (i,j)좌표에 있는 각 화소값을 나타내고, S(i+u,j+v)는 기준블럭의 위치에서 있는 각 화소값을 나타낸다. $ADC_{uv}(i,j)$는 S(i+u,j+v)와 R(i,j)의 절대치 차이 값을 의미한다. 수식(2)에서 사용된 $PS_{u,v}(k,j)$는 (u,v)좌표의 이동벡터를 갖는 탐색블럭의 j 열의 k 행까지의 화소값들과 기준블럭의 j 열의 k 행까지의 화소값들의 각각의 화소값 절대치 차이값 누적치를 나타낸다. 그리고 수식(3)의 MAD(u,v)는 (u,v) 좌표의 이동벡터를 갖는 탐색블럭과 기준블럭과의 절대치 차이 값의 총 누적 값을 나타낸다. 블럭정합 알고리즘은 이렇게 구해진 (u,v)좌표가 -p/+p 범위내의 모든 MAD(u,v) 중에서 가장 최소 값을 갖게 하는 (u,v)를 기준블럭의 이동벡터로 결정한다.

Ⅱ. 완전탐색 블럭정합 알고리즘의 이차원 시스토릭 어레이 구조

1. 전체 시스템 구조

본 논문에서는 완전탐색 블럭정합 알고리즘을 VLSI로 구현하기 위해서 앞에 기술한 수식(1),(2),(3)를 시스토릭 어레이화 시켰다.[7][8] 수식(1)을 구현하기 위해서 절대치 계산이 가능한 처리기(PE : Processing Element,이하 PE라 칭함)를 만든다. 수식(2)을 구현하기 위해서는 PE를 N개 행방향으로 어레이 시키고 수식(3)을 구현하기 위해서는 N개의 행방향으로 PE들을 N개 열방향으로 나열하여 어레이 시키면 N^2 개의 PE들로 완전탐색 블럭정합 알고리즘을 병렬처리하는 VLSI 구조를 구현할 수 있다.[9] 본 논문은 이의 수행을 위해서 종래의 구조[9][10][11][12] 에서 문제시되었던 과다한 동작 사이클을 줄이기 위해서 병렬 파이프라인 처리과정 동안 발생하는 무효한 탐색블럭을 계산하는 동작 사이클을 발생시키지 않는 방법으로 그림 1 과 같은 구조를 갖는 이차원 병렬처리 시스토릭 어레이를 구성하였다.

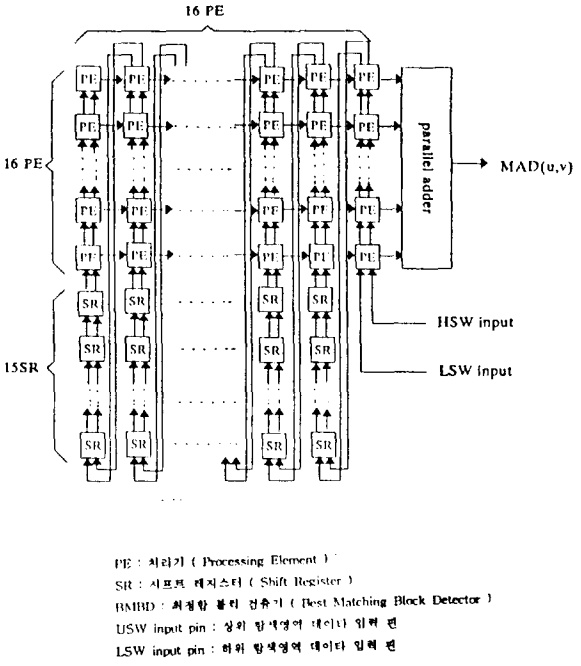


그림 1. 완전탐색 블럭정합 알고리즘의 구현을 위한 2 차원 시스톨릭 어레이 구조

Fig. 1. Two dimensional systolic array architecture for implementation of full search block matching algorithm.

제시된 어레이 구조에서 각각의 PE는 자기 고유의 기준블럭 데이터를 가지고 있어야한다. 그리고 동작 수행 시 매 사이클마다 어레이의 각 PE에 있는 모든 탐색 영역 데이터는 그림 1 에 나타나 있는 바와 같이 한 위치 씩 화살표 방향으로 다음의 PE나 쉬프트 레지스터(shift register : 이하 SR이라 칭함)로 이동한다. 결국 매 사이클마다 제시된 어레이 구조는 다른 이동벡터를 갖는 탐색블럭에 대한 MAD(u,v)를 계산한다. 기준블럭 데이터를 각각의 PE 내부의 레지스터에 저장할 때 SR 을 거치지 않기 위해서 PE 에서 SR로 연결된 배선 중간에 MUX를 달아 그림 1 과 같이 PE 어레이 내부에서만 데이터가 움직이도록 하는 구조를 갖는다.

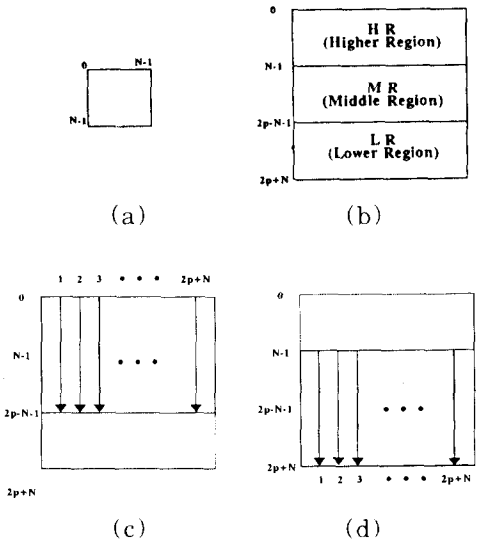


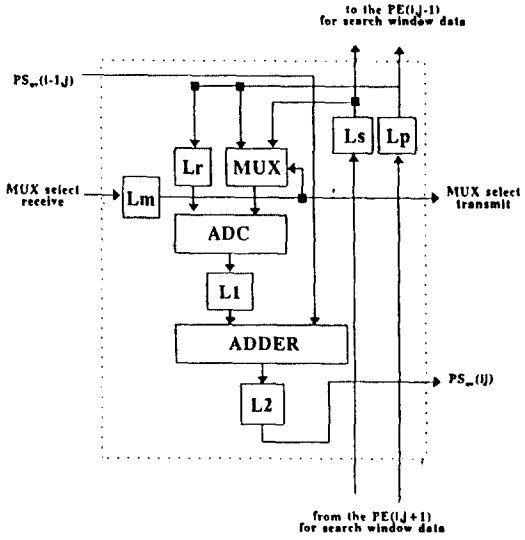
그림 2. 효율적인 탐색영역 구분과 USW 와 LSW 의 탐색영역 데이타 입력열
(a)기준블럭 (b)효율적인 탐색영역 구분
(c)USW 핀 탐색영역 데이타 입력열
(d)LSW 핀 탐색영역 데이타 입력열

Fig. 2. Efficient search window division and input data order of USW and LSW pins.
(a)Reference Block (b)Efficient search window division (c)Input data order for USW pin (d)Input data order for LSW pin

제시된 구조는 탐색거리와 기준블럭의 크기에 따라서 (2p-N-1)x(N-1) 개의 SR 을 사용한다. SR 의 기능은 유효한 탐색블럭을 형성하기 위해서 PE 에서 계산되어야 하는 탐색영역 데이터를 대기시키는 역할을 한다. 이렇게 설계된 칩을 여러개 동시에 사용하여 기준블럭을 키워서 화면 정보량이 많은 경우에 적용할 수 있다. 탐색영역의 데이터 입력 열을 위해서 탐색영역을 그림 2 (b) 와 같이 세 부분으로 나눈다. 탐색영역 중 위 부분부터 기준블럭의 열의 화소 개수와 같은 크기를 갖는 영역을 HR(Higher Region)이라 하고 아래 부분에서부터 기준블럭의 열의 화소 개수와 같은 크기를 갖는 영역을 LR (Lower Region) 이라 하고 탐색영역 중 HR 과 LR 에 속하지 않는 영역을 MR (Middle Region) 이라고 한다. 탐색영역의 데이터 입력 열을 위한 USW 핀 입력 열은 HR 과 MR 의 데이터를 그림 2 (c) 에 나타난 화살표 방향으로 데이터를 읽고 행방향으로 1,2,…,2p+N 의 순서로 순차적으로 입력시킨다. 이렇게 탐색영역을 세 영역으로 구분함으로써 USW 와 LSW 로 MR 영역을 겹쳐서 입력시키는 과부하에도 불구하고 기존의 구조와 마찬가지로 데이터의 흐름을 방해하지 않고 무효한 연산 사이클을 발생시키지 않게 된다.

2. 처리기 내부의 구조

PE(i,j) 내부의 구조는 그림 3 과 같다.



ADC : 절대치 계산기
Lp, Ls, Lr, Lm, L1, L2 : 레지스터

그림 3. 처리기의 내부구조

Fig. 3. Internal architecture of PE.

PE(i, j) 란 (i, j) 좌표에 위치한 처리기를 의미한다. PE 내의 Ls와 Lp 는 각각 탐색영역 데이터인 USW 핀 입력 열과 LSW 핀 입력열의 데이터를 저장하고 있다가 매 사이클마다 위쪽 방향의 PE(i, j-1)로 이동된다. 기준블럭의 전처리 수행 동안 Lp 를 통하여 Lr 로 기준블럭 데이터를 저장한다. 그리고 멀티프렉서(이하 MUX라 칭함) 는 매번 유효한 탐색블럭을 형성하기 위해서 Ls와 Lp에 있는 탐색영역 데이터 중에서 알맞은 탐색영역 데이터를 선택한다. 절대치 계산기(ADC) 의 결과값은 덧셈기의 입력으로 사용되기 위해서 L1 에 저장된다. 그리고 L1의 값은 이후 덧셈기에서 PE(i-1, j) 로 부터 입력된 부분 합과 더해져서 PE(i+1, j)로 넘길 부분 합을 계산한다. 덧셈기에서 계산된 부분합은 다음 단의 PE내의 덧셈기의 입력으로 사용되기 위해 L2 에 저장된다. Lm 은 MUX를 정 동작시키기 위한 선택신호를 PE(i-1, j) 로 부터 입력받아 저장하여 사용하고 다음 사이클에 PE(i+1, j) 로 MUX 선택신호를 전달한다. 전체적으로 PE 는 절대치 계산기와 덧셈기로 2 단계의 파이프라인 구조로 이루어졌다. 그리고 PE(i, j) 내부의 덧셈기의 입력 비트수는 (8+q) 비트이면 되는데 여기에서 q = [log2i] 로 구해진다. 여기에서 i 는 PE(i, j) 에 있는 i 첨자와 같다.

3. BMBD 내부 구조

가능한 모든 MAD 값 중에서 최소의 값을 결정하는 BMBD 는 그림 4 와 같은 구조를 갖는다.

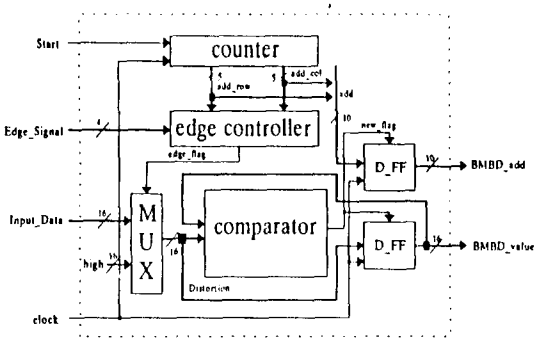


그림 4. BMBD 내부 구조

Fig. 4. Internal architecture of BMBD.

BMBD(Best Matching Block Detector) 는 가능한 모든 탐색블럭들 중에서 기준블럭과의 MAD 값이 가장 작은 탐색블럭을 찾는 기능을 한다. 어레이 프로세서에 순차적으로 각 후보블럭의 MAD 값들이 한 클럭마다 BMBD 에 입력된다. BMBD 외부 제어기에서는 BMBD 내부의 카운터를 동작시켜서 현재 들어오는 MAD 값이 어떤 이동벡터를 갖는 탐색블럭의 MAD 인지 주소를 발생시킨다. 즉 BMBD 내부의 카운터는 간단한 주소 발생기이다. BMBD 의 또 하나의 입력은 Edge_sig 이다. 이건 현재 계산 수행중인 기준블럭의 프레임 내에서의 위치에 대한 정보이다. 즉 기준블럭의 프레임 내부에서의 위치에 따라 탐색영역의 데이터가 프레임 메모리에 속하지 않는 경우의 탐색블럭에 대하여 BMBD 내부에서 강제적으로 어레이로부터 어떤 값이 입력되더라도 그 값은 무시하고 강제적인 값을 비교기에 입력시켜준다. 이러한 결정을 하는 부분이 edge_controller 이다. 여기에서 발생되는 edge_flag 가 high 인 경우는 Input_Data 는 무시되고 강제적인 값을 비교기에 입력시킨다. 그리고 BMBD 의 F/F 은 모든 탐색블럭 중에서 가장 작은 MAD 값을 갖는 블럭의 MAD 값과 그의 주소를 저장하는 역할을 한다. 비교기에서 어레이로부터 입력되는 MAD 값은 F/F 에 저장하고 있는 값과 비교하여 입력된 MAD 값이 더 적으면 비교기는 new_flag 를 high 로 만든다. 비교기에서 new_flag 가 high 되어 F/F 을 enable 시키면 그 때의 MAD 값과 그의 주소가 새로운 이동벡터에 해당하는 MAD 값과 이동벡터로 저장된다.

4. MUX 제어신호

어레이 프로세서가 정 동작하기 위해서는 PE내의 MUX의 선택신호는 다음과 같이 동작하여야 한다. 처리기 PE(1, j)에서 유효한 탐색영역의 데이터를 Ls와 Lp 데이타 중에서 선택하기 위한 MUX 제어로직은 그림 5 와 같이 동작해야한다.

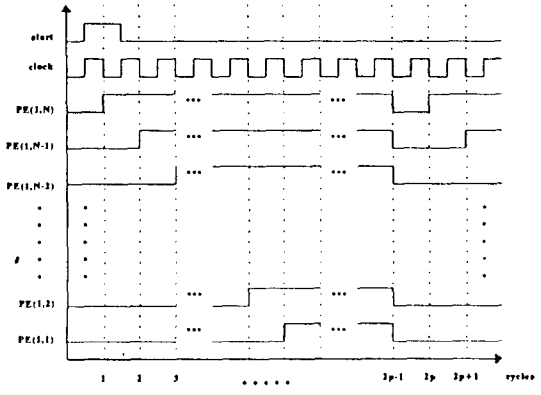


그림 5. 멀티프렉스 제어신호

Fig. 5. Control logic for multiplexer.

전처리 수행 후 처음 두 사이클 동안은 PE(i, 1) 은 Lp

의 데이터를 선택하지 않고 Ls 의 데이터를 선택한다. 전처리 수행 후 세번째 사이클에는 PE(1,N)과 PE(1,N-1)이 Lp 의 데이터를 선택한다. 이와 같이 한 사이클에 하나의 PE 가 더 Lp 의 데이터를 선택하다가 k 번째 사이클에서는 PE(1,N),…,PE(1,N-k+2) 이 Lp 의 데이터를 선택한다. 결국 N+1 사이클에는 모든 PE 가 Lp 의 데이터를 선택한다. 그리고 2p-N-1 사이클 동안 모든 PE 가 Lp 의 데이터를 선택하고 2p-1 의 사이클에는 MUX 제어신호는 전처리 수행 후 첫 사이클과 동일한 상태로 돌아간다. 그리고 이 동작은 처리동작 수행동안 같은 방법으로 계속되어질 것이다. 또한 각각의 PE(i,j) 는 PE 에 있는 Lm 을 통하여 PE(i,j-1) 로 부터 한 사이클 지연된 신호를 MUX의 선택 신호로 받는다.

Ⅲ. 시스템 동작 설명

제시된 새로운 이차원 어레이 구조의 동작 예를 보이기 위해 그림 6 과 같은 기준블럭의 크기 (N×N) 가 3×3 이고, 탐색길이 (p) 의 크기가 3 인 경우에 대한 동작 예를 다음에 보였다.

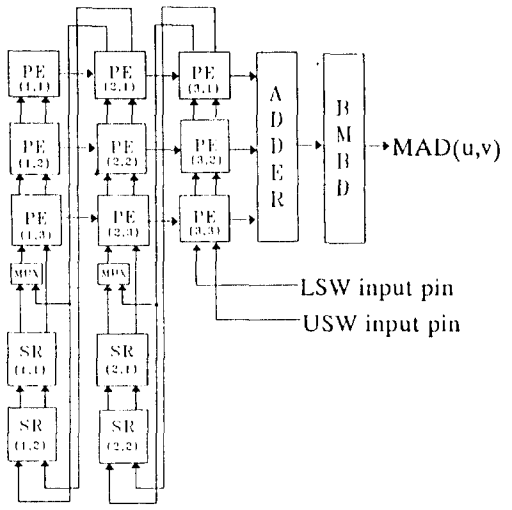


그림 6. NxN = 3x3, p = 3 인 경우의 어레이 시스템 구성

Fig. 6. System array in case NxN = 3x3, p = 3.

수행동작의 흐름도는 표 [1] 에 나타나 있다. 표 [1] 에 있는 [S(i,j) S(k,l)] 중에서 앞의 S(i,j)는 PE 내부에 있는 Ls 에 들어있는 탐색영역 데이터를 의미하며 뒤의 S(k,l) 은 Lp 에 들어있는 탐색영역 데이터를 의미한다. 또한 밑줄 친 S(i,j) 유효한 탐색블럭을 형성하기 위해 MUX에 의해 선택된 탐색영역 데이터를 의미한다. MEP가 정 동작하기 위해서는 기준블럭 데이터는 9 사이클 (N×N) 의 전처리 수행 동안에 Lr 로 저장되어서 하나의 기준블럭에 대한 이동벡터가 구해질 때까지 저장되어 있다. 기준블럭 데이터의 전처리 수행 이후 PE 와 SR 에 각각 탐색영역의 데이터를 또한 12 사이클 (2p×(N - 1)) 동안 전처리 수행이 필요하다. 그러면 사이클 22 에 표 [1] 에 나타난 것과 같이 처리동작수행을 위해 모든 레지스터가 준비된다. 그리고 사이클 22 에 PE(1,j) (1≤j ≤3 : 이하 첨자 j 를 사용하는 경우는 1 부터 3 까지 인 PE 각각의 경우를 나열한 것으로 의미한다.) 는 $PS_{3,-3}(1,j)$ 을 계산하고 PE(2,j) 와 PE(3,j) 은 유효한 계산을 하지는 않는다.

표 1. 본 논문이 제시한 신호처리흐름도의 구현예 (기준 블럭이 3×3 이고 탐색 거리가 3 인 경우)

Table 1. Implementation example of the signel processing flow proposed by the paper. (reference block size is 3×3, and search displacement is 3)

클럭 사이클	입력 데이타열과 각 처리기의 탐색영역 데이타와 덧셈기의 입력과 출력 — 처리기 어레이			덧셈기입력	덧셈기출력
	PE(1,1)	PE(2,1)	PE(3,1)		
	PE(1,2)	PE(2,2)	PE(3,2)		
	PE(1,3)	PE(2,3)	PE(3,3)		
	SR(1,1)	SR(2,1)			
	SR(1,2)	SR(2,2)			
1~21	전 처 리 수 행			x x x	x
22	[S(1,1)S(x,x)]	[S(1,6)S(1,6)]	[S(2,5)S(2,5)]	x	
	[S(1,2)S(x,x)]	[S(2,1)S(1,7)]	[S(2,6)S(2,6)]	x	
	[S(1,3)S(x,x)]	[S(2,2)S(1,8)]	[S(3,1)S(2,7)]	x	x
	[S(1,4)S(1,4)]	[S(2,3)S(1,9)]			
	[S(1,5)S(1,5)]	[S(2,4)S(2,4)]			
23	[S(1,2)S(x,x)]	[S(2,1)S(1,7)]	[S(2,6)S(2,6)]	x	
	[S(1,3)S(x,x)]	[S(2,2)S(1,8)]	[S(3,1)S(2,7)]	x	
	[S(1,4)S(1,4)]	[S(2,3)S(1,9)]	[S(3,2)S(2,8)]	x	x
	[S(1,5)S(1,5)]	[S(2,4)S(2,4)]			
	[S(1,6)S(1,6)]	[S(2,5)S(2,5)]			
24	[S(1,3)S(x,x)]	[S(2,2)S(1,8)]	[S(3,1)S(2,7)]	$PS^1(-3,-3)$	
	[S(1,4)S(1,4)]	[S(2,3)S(1,9)]	[S(3,2)S(2,8)]	$PS^2(-3,-3)$	
	[S(1,5)S(1,5)]	[S(2,4)S(2,4)]	[S(3,3)S(2,9)]	$PS^3(-3,-3)$	MAD(-3,-3)
	[S(1,6)S(1,6)]	[S(2,5)S(2,5)]			
	[S(2,1)S(1,7)]	[S(2,6)S(2,6)]			
⋮	⋮	⋮	⋮	⋮	⋮
65	[S(7,6)S(7,6)]	[S(8,5)S(8,5)]	[S(9,4)S(9,4)]	$PS^1(3,0)$	
	[S(8,1)S(7,7)]	[S(8,6)S(8,6)]	[S(9,5)S(9,5)]	$PS^2(3,0)$	
	[S(8,2)S(7,8)]	[S(9,1)S(8,7)]	[S(9,6)S(9,6)]	$PS^3(3,0)$	MAD(3,0)
	[S(8,3)S(7,9)]	[S(9,2)S(8,8)]			
	[S(8,4)S(8,4)]	[S(9,3)S(8,9)]			
66	[S(8,1)S(7,7)]	[S(8,6)S(8,6)]	[S(9,5)S(9,5)]	$PS^1(3,1)$	
	[S(8,2)S(7,8)]	[S(9,1)S(8,7)]	[S(9,6)S(9,6)]	$PS^2(3,1)$	
	[S(8,3)S(7,9)]	[S(9,2)S(8,8)]	[S(x,x)S(9,7)]	$PS^3(3,1)$	MAD(3,1)
	[S(8,4)S(8,4)]	[S(9,3)S(8,9)]			
	[S(8,5)S(8,5)]	[S(9,4)S(9,4)]			
67	[S(8,2)S(7,8)]	[S(9,1)S(8,7)]	[S(9,6)S(9,6)]	$PS^1(3,2)$	
	[S(8,3)S(7,9)]	[S(9,2)S(8,8)]	[S(x,x)S(9,7)]	$PS^2(3,2)$	
	[S(8,4)S(8,4)]	[S(9,3)S(8,9)]	[S(x,x)S(9,8)]	$PS^3(3,2)$	MAD(3,2)
	[S(8,5)S(8,5)]	[S(9,4)S(9,4)]			
	[S(8,6)S(8,6)]	[S(9,5)S(9,5)]			

사이클 23 에는 PE 내의 Lp 와 Ls 의 탐색영역 데이터는 그림 6 에 있는 화살표 방향으로 각각의 PE 들과 SR 로 탐색영역의 데이터가 이동되고 각각의 PE(1,j) 는 $PS_{3,-2}(1,j)$ 을 계산한다. PE(2,j) 는 PE(1,j) 로 부터 $PS_{3,-3}(1,j)$ 을 PE 내의 덧셈기에

입력받아서 $PS_{-3,-3}(2,j)$ 을 계산한다. 그리고 PE(3,j) 은 유효한 계산을 하지 않는다. 사이클 24 에는 역시 탐색영역 데이터는 다른 PE 와 SR 로 이동하고 PE(1,j) 는 $PS_{-3,-1}(1,j)$ 을 계산한다. 그리고 PE(2,j) 는 PE(1,j) 로 부터 $PS_{-3,-2}(1,j)$ 을 입력받아서 $PS_{-3,-2}(2,j)$ 을 계산한다. 그리고 PE(3,j) 은 PE(2,j) 로 부터 $PS_{-3,-3}(2,j)$ 을 입력받아서 $PS_{-3,-3}(3,j)$ 을 각각 계산한다. 사이클 25 에는 또한 모든 PE 와 SR 의 탐색영역 데이타는 다른 PE 와 SR로 이동되고 PE(1,j) 은 $PS_{-3,0}(1,j)$ 을 계산한다. PE(2,j) 는 PE(1,j) 로 부터 $PS_{-3,-1}(1,j)$ 을 입력받아 $PS_{-3,-1}(2,j)$ 을 계산한다. PE(3,j) 은 PE(2,j) 로 부터 $PS_{-3,-2}(2,j)$ 을 입력받아 $PS_{-3,-2}(3,j)$ 을 계산한다. 그리고 병렬 덧셈기는 수식 (3) 과 같은 동작을 수행하여 MAD(-3,-3) 를 계산한다. 사이클 26 에서 다른 모든 동작은 계속되고 특히 MAD(-3,-2) 가 병렬 덧셈기에서 출력된다. 계속되는 이러한 동작들로 모든 MAD(u,v) 는 병렬 파이프라인 동작으로 매 사이클마다 계산된다.

Ⅳ. 성능 분석

N×N 크기의 기준블럭과 탐색거리가 p 인 경우 기준블럭의 전처리 수행을 위해서 N^2 의 사이클이 소요되고 USW 핀과 LSW 핀으로 탐색영역의 데이터를 전처리 수행하는데는 (N-1)×(2p) + 1 의 사이클이 소요된다. 그리고 MAD(u,v) 의 계산을 위해서는 가능한 후보 탐색블럭의 개수와 같은 $(2p)^2$ 만큼의 사이클이 소요된다. 결국 한 기준블럭에 대해서 요구되는 사이클은 $N^2 + (N-1)\times(2p) + 1 + (2p)^2$ 이다. 제시된 구조의 임계 속도는 하나의 덧셈기와 플립플롭의 지연시간의 합이다. 이것은 VTI 0.8um CMOS 기술로 각각 11.3 ns 와 2.3 ns 이다. 결국 제시된 구조에 요구되는 클럭 주기는 13.9 ns 이다. 표 [2] 에 MPEG-2 에서 요구하는 기준에 본 논문에서 제시된 구조가 가질 수 있는 최대 프레임 레이트가 나타나 있다.[13] 즉 MPEG-2 의 main profile 의 Low Level 의 경우인 352×288 의 화면 크기에 대해 8×8 의 기준블럭과 탐색 거리가 8 인 경우에는 MPEG-2 가 요구하는 30 frame / sec 보다 300 % 높은 처리 속도를 나타냈다. 그리고 main level 의 경우인 720×486 의 화면 크기에 대해서는 30.38 frame / sec 로써 기준 (30 frames / sec) 을 만족한다는 것을 알 수 있다.

본 논문에서 새로이 제시한 2 차원 시스토릭 어레이의 경우 N×N 의 기준블럭과 p 의 탐색 영역 거리를 갖는 시스템인 경우에 있어서 요구되는 PE 의 개수는 N^2 개이고 요구되는 SR 의 개수는 (2p-1)×(N-1) 이다.

표 2. MPEG-2 기준에 맞춘 제안된 구조의 최대 프레임 레이트

Table 2. Maximum frame rate of the proposed architecture satisfying the MPEG2 standard.

standard	MPEG-2 standard	
	LOW level	MAIN level
화면 크기 (수평화소 × 수직화소)	352 × 288	720 × 486
기준블럭 크기 (수평화소 × 수직화소)	8 × 8	8 × 8
탐색 영역	-8 / +7	-8 / +7
최대 프레임 레이트 (frame / sec)	104 . 89	30 . 38

표 3. 기존의 2 차원 구조와 새로이 제시한 움직임 추출기의 성능 비교
(N×N = 8×8, p = 8 인 경우의 비교)

Table 3. Performance Comparison of existing two-dimensional architectures with the newly-proposed one. (when N×N is 8×8, and p=8)

	V. & Stegher	Hsieh & Lin	STI-3220	제안된 구조
처리기 갯수	64	64	256	64
주변 레지스터갯수	256	105	112	49
한기준블럭에 필요한 사이클	417	577	230	369
데이터칩의 복잡성	HIGH	MIDDLE	LOW	LOW
처리기의 복잡성	LOW	LOW	MIDDLE	LOW

일차원 시스토릭 어레이 구조를 갖는 LSI - LOGIC 사의 L64720 과 비교해서 처리 속도 면에서는 월등히 뛰어나며 같은 2차원 시스토릭 어레이 구조를 사용하는 나온 SGS - THOMSON 사에서 제작한 구조와 비교할 때 동작 사이클 면에서는 N = 8, p = 8 인 경우 60 % 만큼 동작 사이클이 더 걸리지만 칩 면적에서 우선 처리기의 개수만을 비교할 때 25 % 만 있으면 가능하고 주변 레지스터의 경우는 43 % 만 있으면 되므로 훨씬 경제적인 것을 알 수 있다. [11]-[12] 높은 데이터 압축과 고속 처리를 요하는 MPEG-2 의 main level 의 응용을 위하여 움직임 벡터 추정기 설계할 때 칩 면적은 아주 중요한 요소중 하나이다.

Ⅴ. 결 론

본 논문은 효율적인 탐색영역의 분할로부터 얻어진 이차원 어레이 구조에 기반을 둔 완전탐색에 의한 블럭정합 알고리즘을 위한 집적회로 구조를 보였다. 파이프라인과 병렬처리의 특성을 잘 이용한 효율적인 처리기의 설계와 처리기들의 시스토릭 어레이 구조로 시스템 클럭 주파수를 높였다. 기존의 구조에서 문제시되었던 점들을 제시된 구조는 효율적인 탐색영역의 분할로 무효한 탐색블럭을 형성하는 사이클을 없앴고 하드웨어 구조 면에서도 다른 구조 보다 더 적은 면적을 차지하고 집적회로의 구현이 용이하도록 모듈성과 규칙성에 입각하여 설계되었다. 그리고 제시된 구조는 MPEG-2 의 응용에 적합하다.

參 考 文 獻

[1] 백종섭,이용훈,남승현,이문기, "Pel Recursive Algorithm 의 ASIC 구현을 위한 화소값 기울기 발생기 설계", 대한 전자공학회 1992년도 추계종합학술 대회 논문집 제15권, 제2호, pp.604-606, 1992

[2] J.S. Baek, S.H. Nam and M.K Lee, "VLSI Implementation of full search block matching algorithm", JTC-CSCC '93, vol.2-2, pp. 909-912, Nara, July 1993.

[3] J.S. Baek, S.H. Nam and M.K Lee, "A Fast Array Architecture for Block Matching Algorithm", ISCAS'94, pp. 211-214, London, May 1994.

[4] 이용훈, 백종섭, 남승현, 이문기, "3 단계 검색 블럭 정합 알고리즘의 VLSI 처리기 설계", 대한 전자공학회 1992년도 추계종합학술대회 15권, 2호, pp.601-603, 1992

[5] 백종섭, 남승현, 이문기, "완전 탐색에 의한 움직임 추정기 시스토릭 어레이 구조", 전자정보통신논문집, 전자 정보 통신 연구소, 제1권, 제1호, 1994.

[6] H. T. Kung, "Why Systolic Architecture", IEEE trans. on computer, Jan. 1982.

[7] S.Y.Kung, "VLSI Array Processor", prentice hall, Eagle wood cliffs, 1998.

[8] T.Komarek and P.Pirsh, "Array architectures for block matching algorithms," IEEE Trans. on Circuits and Systems, vol.36, no.10, pp.1301-1308, Oct. 1989.

[9] C.H. Hsieh, T.P. Lin, "VLSI architecture for block matching motion estimation algorithm," IEEE Trans. on Cir. & Sys. for video Tech., pp.169-175, June 1992.

[10] K.M. Yang, M.T. Sun and L. Wu, "A Family of VLSI Design for the Motion Compensation Block Matching Algorithm", IEEE Trans. on Circuits and Systems, vol.36, No. 10, pp. 1317-1325, October 1989.

[11] A. Artieri, F. Jutand, "A Versatile and Powerful Chip for Real Time Motion Estimation", IEEE, pp. 2453-2457, 1989.

[12] Didier Le Gall, "MPEG:A Video Compression Standard for Multimedia Applications", Commui. of the ACM, vol.34, no.4, pp.46-59, April 1991.

著者紹介

白 鍾 燮(正會員)

1968年 11月 22日生. 1992年 2月 서울대 전자공학과(학사). 1994年 6月 연세대학교 전자공학과(석사). 1994年 7月 ~ 현재 현대전자 (주) 반도체 연구소. 주관심 분야는 디지탈신호처리 VLSI 설계, 영상압축 알고리즘 연구 및 VLSI 구현 등임.

•

李 文 基(正會員)

1941年 8月 23日生. 1973年 연세대학교 공학박사. 1973年 경희대학교 부교수. 1980年 University of Oklahoma, Ph.D. 1980年 한국전자기술연구소(현 ETRI) 책임연구원. 1982年 ~ 현재 연세대학교 전자공학과 교수. 1992年 ~ 현재 연세대학교 아식설계공동연구소 소장. 1993年 ~ 현재 대한 전자공학회 부회장. 주관심 분야는 VLSI & CAD 및 ASIC 설계 등임.

南 承 鉉(正會員)

1963年 11月 5日生. 1986年 2月 연세대학교 전자공학과(학사). 1988年 2月 연세대학교 전자공학과(석사). 1988年 ~ 1990年 대우통신(주). 1990年 ~ 현재 대우(주). 1991年 ~ 현재 연세대학교 전자공학과(박사과정). 주관심 분야는 디지탈신호처리 VLSI 설계, 영상압축알고리즘 연구 및 VLSI 구현 등임.