論文 92-29B-10-2

고속 영상처리를 위한 다중접근 기억장치의 구현

(An Implementation of Multiple Access Memory System for High Speed Image Processing)

金 洁 潤*, 李 興 揆**, 朴 宗 元***

(Gil Yoon Kim, Heung Kyu Lee, and Jong Won Park)

要 約

본 논문에서는 고속 영상처리를 위한 기억구조를 구현하였다. 이 기억구조는 p와 q를 제작인수로 하여 이차원 영상 평면내의 블럭(p×q), 수평벡터(1×pq) 그리고 수직벡터(pq×1)내의 여러 영상점들을 동시에 접근한다. 이 기억 구조는 주소계산회로, 주소 재배열회로, 자료 재배열회로, 모듈선택회로 그리고 m>pq인 m개의 기억모듈로 구성한다. 이 주소계산회로는 영상점들의 주소들 사이의 상관관계를 이용하여 pq개의 주소들을 동시에 계산한다. 본 논문에서는 모듈선택회로에 재배열회로를 확장 적용하여 별도의 모듈 할당회로가 사용되지 않도록 개선하였으며, 로직 시뮬레이터 Verilog-XL로 시뮬레이션하여 설계한 회로를 검증하고 성능을 평가하였다. 구현한 시스템은 16개의 영상점을 동시에 접근할 수 있으며 보통의 기억장치보다 6배 이상의 처리속도를 갖는다.

Abstract

This paper considers an implementation of the memory system which provides simultaneous access to pq image points of block(p×q), horizontal vector(1×pq)and/vertical vector(pq×1) in 2-dimension image array, where p and q are design parameters. This memory system consists of an address calculation circuit, address routing circuit, data routing circuit, module selection circuit and m memory modules where m>pq. The address calculation circuit computes pq addresses in parallel by using the difference of addresses among image points. Extra module assignment circuit is not used by improving module selection circuit with routhing circuit. By using Verilog-XL logic simulator, we verify the correctness of the memory system and estimate the performance. The implemented system provides simultaneous access to 16 image points and is 6 times faster than conventional memory system.

I. 서 론

대량의 영상자료를 고속으로 처리하기 위해서는 먼저 영상점들에 대한 접근시간을 줄여야한다. 그러나 영상자료의 크기가 대단히 크기때문에 접근시간이 빠른 고가의 메모리를 영상기억장치로 사용하기는 어렵다. 그런데 보통 영상처리에서 많이 사용되는 연산은 개개의 영상점을 대상으로 하기보다는 특정한 형태(이를테면, 2차원 블럭, 수평벡터, 수직벡터 등)의 영상점군에 대한 연산이 대부분이다. 그래서 이러한 여러개의 영상점군에 대하여 동시에 접근이 가능하도록 기억장치를 설계하여 영상점에 대한 접근시간을 줄이려는 많은 연구가 있었다.[1,3,5,6,7]

*準會員 **正會員, 韓國科學技術院 電算學科
(Dept. of Computer Science, KAIST)
***正會員, 忠南大學校 電算學科
(Dept. of Computer Science, Chungnam Nat'l Univ.)
接受日字 : 1992年 3月 17日

최근 영상처리를 위한 효율적인 변환 기법으로 각광을 받고 있는 wavelet 변환을 고속으로 처리하기 위해서도 이러한 다중접근기억장치는 효과적으로 사용된다. Wavelet 변환에 의한 다중 해상도 표현기법에서는 2차원 영상에 대하여 가로방향과 세로방향의 두 직교 기저에 대하여 각각 1차원 wavelet 변환을 행하는 방법을 사용한다. 그러므로 여러개의 영상점을 동시에 변환하기 위해서는 가로벡터와 세로벡터에 대한 동시 접근이 필요하다. 또한 변환된 데이타에 대한 압축 또는 인식 알고리즘을 효과적으로 수행할 수 있도록 2차원 블럭형태에 대해서도 동시 접근이 가능해야 한다.

여러가지 형태의 접근집합에 대하여 동시접근 가능한 기억구조에 관한 연구는 각 접근집합을 동시에 접근할 수 있도록 메모리 모듈에 분산시키는 방법에 관한 연구, 각 자료들의 주소를 결정하는 주소생성 회로에 관한 연구[3, 5, 6, 7] 그리고 각 모듈에 저장된 자료를 원 영상행렬의 순서로 조정하는 재배열 회로에 관한 연구[2, 5, 6, 7]로 나눌 수 있다.

이중 특히 Park[5]은 주소생성회로를 주소계산회로와 주소이동회로로 분리하여 주소생성회로에서 모듈러 연산을 제거 하였으며, 영상점들 사이의 관계를 이용한 효율적인 주소계산회로를 제시함으로써 pq개의 주소를 동시에 계산하여 주소생성회로를 단순화 시켰다. Park[6, 7]에서는 [5]의 주소계산회로에서 접근 형태에 따라 계산시간이 다르고, 재배열회로에서 수직벡터의 회전수가 블럭·수평벡터의 경우와 다른 두가지 문제점을 해결하였다. 또한 주소계산회로를 대각선과 역대각선에 확장 적용하였으며 파이프라인 기법을 사용하여 임의 각도의 벡터에 대하여 동시접근 가능하도록 기억 구조를 개선하였다.

그러나 이러한 지금까지의 연구는 메모리 시스템을 구성하는 각 부분 회로를 중심으로 이루어졌으며, 전체 시스템 측면에서의 고려는 부족한 실정이다. 영상 처리용 메모리 시스템은 각 부분 회로들의 통합 시스템이기 때문에 단순히 각 부분회로를 합해서는 많은 부분의 회로가 중복되므로, 전체 시스템의 관점에 각 회로간의 중복을 줄일 수 있도록 부분 회로를 개선 해야한다. 이 때 특정한 부분 회로의 개선보다는 각 회로간의 균형이 중요한 요소가 되므로 이를 고려해서 설계 해야한다.

본 논문에서는 이러한 전체 시스템 측면에서의 성능향상과 회로의 단순화에 중점을 두었다. 이를 위하여 기억모듈 선택회로에 재배열회로를 확장 적용하여 별도의 모듈러 연산을 사용하지 않도록 하였으며, 속도와 chip면적을 고려하여 각 계산경로간의 지연시간차를 줄임으로써 회로를 최적화 하였다. 본 논문은 고속 wavelet 변환 시스템[8]의 영상 기억장치로 연구 되었으므로 wavelet 변환에서 많이 사용되는 블럭, 가로벡터, 세로벡터 세 가지 접근 형태만을 고려하였으며, 대각선, 역대각선 등의 접근 형태는 고려하지 않았다.

개선된 다중접근 기억장치는 하드웨어 기술언어인 Verilog로 구현하였으며, 로직 시뮬레이터 Verilog-XL로 검증하고 성능을 평가하였다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 다중접근 기억구조의 개념과 이를 위한 일반적인 설계를 보인다. Ⅲ장에서는 이러한 개념적인 설계를 실제 구현 하기위한 문제점을 살펴보고 이를 고려한 다중접근 기억장치를 설계한다. Ⅳ장에서는 설계한 다중접근 기억장치의 시뮬레이션 결과를 보이고 성능을 평가한다. 결론 및 향후 연구과제에 대하여는 Ⅴ장에서 기술하였다.

Ⅱ. 다중접근 기억장치

영상은 영상점 I(.,.)들의 M×N행렬로 나타낼 수 있는데, 이 때 각 요소 I(i, j)는 $0 \le i \le M-1$과 $0 \le j \le N-1$에 대해 해당부분의 색과 밝기를 나타낸다. 임의의 정수 p, q를 설계상수로 했을 때, pq개의 영상점을 가로벡터(1×pq), 세로벡터(pq×1), 2차원 블럭(p×q)의 세가지 형태로 접근할 때 영상행렬 I(.,.)에서 각각의 접근형태에 따라 동시에 접근되는 영상점들은 다음과 같이 나타낼 수 있다.

블　　럭 :

$$BL(i,j) = \{I(i+a, j+b) \mid 0 \le a < p,\ 0 \le b < q\},\quad 0 \le i \le M-p,\ 0 \le j \le N-q$$

수평벡터 :

$$HS(i,j) = \{I(i, j+b) \mid 0 \le b < pq\},\quad 0 \le i \le M,\ 0 \le j \le N-pq$$

수직벡터 :

$$VS(i,j) = \{I(i+a, j) \mid 0 \le a < pq\},\quad 0 \le i \le M-pq,\ 0 \le j < N$$

이는 그림 1과 같이 나타낼 수 있다.

다음 절부터는 이러한 여러가지 접근 형태에 대하여 동시에 접근이 가능하도록 기억모듈을 배치하는 모듈할당함수와 배치된 모듈내의 주소를 결정하는 주소할당함수에 대하여 알아보고 이러한 두 기능함수를 이용하여 어떻게 다중접근 기억장치가 구성되는지를 살펴본다.

1. 기능 함수

1) 모듈할당함수(μ)

동시에 접근되는 영상점들의 집합을 접근집합이라 하는데, 이 접근집합 S를 동시에 읽거나 쓰기 위해서는, S의 각 원소들이 서로 다른 모듈에 있어야 한다. 메모리 모듈의 수를 m이라 하면 영상행렬의 각 원소는 m개 중

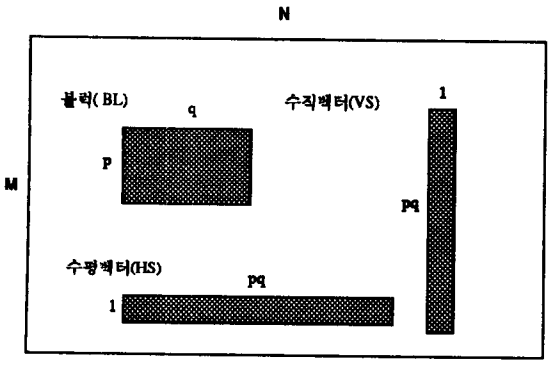


그림 1. 접근 형태
Fig. 1. Access pattern.

한 모듈에 있게 된다. 이렇게 각 영상점이 위치할 모듈을 결정하는 함수 μ를 모듈할당함수(module assignment function)라 하며 다음과 같이 정의한다.

$$\mu : I(i, j) \Rightarrow c (0 \leq c < m),\ 0 \leq i < m,\ 0 \leq j < N$$

접근집합 S의 각원소를 동시에 접근 하기 위해 μ는 다음과 같은 성질을 가져야 한다.

$$(i_1, j_1), (i_2, j_2) \in S \Rightarrow \mu(i_1, j_1) \neq \mu(i_2, j_2)$$

m이 pq보다 큰 경우 m−pq개의 기억모듈이 사용되지 않기 때문에 기억장치의 효율적인 이용측면에서 점에서 m=pq인 경우가 이상적이다. 그러나 이 경우에 동시에 접근 가능한 접근 형태는 가로벡터, 세로벡터, 블럭형태의 일부분에 대해서만 가능하고 이 세가지 형태 모두에 대하여 동시에 메모리에 접근 할 수 있는 μ는 존재하지 않는다.[3] 그러므로 세가지 형태 전부에 대하여 기억장치에 동시 접근하기 위해서는 m은 pq보다 적어도 하나가 많아야 한다. 즉, m≥pq+1의 관계가 성립한다.

μ는 보통 스큐(skew)함수[1, 2, 3, 4]를 사용하여 얻어지며, 다음 식과 같은 일반형을 갖는다.(x//y는 정수 x를 정수 y로 나눈 나머지를 나타낸다.)

$$\mu(i, j) = (\sigma(i, j) = j) // m$$

(m은 모듈갯수, $\sigma(i, j)$는 스큐함수이다)

스큐함수에 따라 여러가지 μ함수가 있으나, $\mu(i, j) = (iq+j) // m$을 사용하면 블럭접근과 수평벡터 접근에 있어 두 접근집합 S간의 모듈번호가 일치하므로 블럭형태에 대한 회로를 줄일 수 있는 장점이 있다.[3]

예를 들어 M=10, N=15, p=2, q=3, m=pq+1, $\mu(i, j) = (iq+j) // m$일 때 영상행렬 I(.,.)의 각 원소는 그림 2와 같다. 임의의 좌표를 기준한 2×3블럭, 크기가 6인 수평 또는 수직 벡터에 대해서도 중복되는 모듈이 없음을 알 수 있다.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0
1	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3
2	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2
4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5
5	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1
6	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4
7	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0
8	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3
9	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6

그림 2. 모듈할당함수의 예
Fig. 2. A sample module assignment function.

2) 주소할당함수(α)

접근집합 S의 각 영상점에 대응하는 모듈은 μ함수에 의하여 결정되지만, 실제로 그 모듈에 위치하기 위해서는 모듈내 주소도 같이 정해져야 한다. 즉 기억장치를 구성하는 관점에서 볼 때 각 영상점은 모듈번호와 모듈내 주소의 쌍으로 접근된다. 따라서 S의 각 원소에 대하여 μ가 가리키는 모듈내의 주소를 결정짓는 함수 α가 필요하다. 이를 주소할당함수(address assignment function)라 하며 다음과 같이 정의된다.

$$\alpha : I(i, j) \Rightarrow k,\ (0 \leq k < \text{각 모듈의 기억용량})$$

α는 μ와 함께 다음 조건을 만족해야 한다.

$$(i_1, j_1) \neq (i_2, j_2)\ and\ \mu(i_1, j_1) = \mu(i_2, j_2) \Rightarrow \alpha(i_1, j_1) \neq \alpha(i_2, j_2)$$

여러가지 α함수가 연구되었으나 다음 한가지에 대해서만 고려하기로 한다.[3, 5] ($\lfloor x \rfloor$는 x보다 작거나 같은 정수중에서 가장 큰 수를 나타내고, $\lceil x \rceil$는 x보다 크거나 같은 정수중에서 가장 작은 수를 나타낸다.)

$$\alpha(i, j) = \lfloor i/p \rfloor s + \lfloor j/q \rfloor$$

여기서 s는 다음 조건을 만족하는 임의의 정수이다.

$s \geq \lceil N/q \rceil, (sM/p \leq$ 각 모듈의 기억용량)

위 α함수를 살펴보면 모듈번호와는 관계없이, 단지 영상행렬내의 위치(i, j)에 의하여 주소가 결정됨을 알 수 있다. 즉 α함수와 μ함수는 서로 독립적이며, 두 함수의 조합으로 접근하려는 영상점이 위치하는 기억장소를 유일하게 지정 할 수 있다.

예를 들어 M=10, N=15, p=2, q=3, $\alpha(i, j) = \lfloor i/p \rfloor + \lfloor j/q \rfloor$일 때 영상행렬 I(.,.)의 각 원소는 그림 3과 같다.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	0	0	0	1	1	1	2	2	2	3	3	3	4	4	4
1	0	0	0	1	1	1	2	2	2	3	3	3	4	4	4
2	5	5	5	6	6	6	7	7	7	8	8	8	9	9	9
3	5	5	5	6	6	6	7	7	7	8	8	8	9	9	9
4	10	10	10	11	11	11	12	12	12	13	13	13	14	14	14
5	10	10	10	11	11	11	12	12	12	13	13	13	14	14	14
6	15	15	15	16	16	16	17	17	17	18	18	18	19	19	19
7	15	15	15	16	16	16	17	17	17	18	18	18	19	19	19
8	20	20	20	21	21	21	22	22	22	23	23	23	24	24	24
9	20	20	20	21	21	21	22	22	22	23	23	23	24	24	24

그림 3. 주소할당함수의 예
Fig. 3. A sample address assignment function.

2. 다중접근 기억장치의 기능 회로

다중접근 기억장치의 구조는 그림 4와 같다. 다중접근 기억장치의 회로는 기억모듈의 주소를 생성하는 주소생성회로, 영상 행렬내의 위치와 실제 모듈의 위치를 일치시키는 재배열회로, 실제 사용되는 모듈을 선택하는 기억모듈 선택회로, 그리고 실제로 자료가 저장되는 기억모듈로 나눌 수 있다.

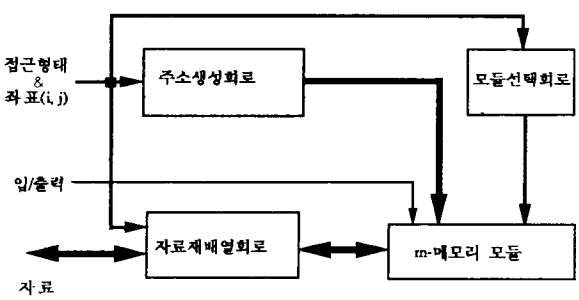


그림 4. 다중접근 기억장치
Fig. 4. Multiple access memory.

1) 주소생성회로

주소생성회로는 모듈할당함수(μ)와 주소할당함수(α)를 사용하여 이루어진다.

I(i, j)와 접근형태가 결정되면 각 모듈에서는 접근집합 S의 한 원소를 갖게 되며, 주소생성회로는 그 원소가 위치하는 모듈내의 물리적 주소를 생성하게 된다.

좌표(i, j)를 기준한 pq개의 영상점에 대한 주소는 각 영상점에 대하여 pq개의 모듈할당회로와 주소할당회로를 이용하여 쉽게 계산 할 수 있으나 회로가 지나치게 복잡해서 현실성이 없다.

Park[5]이 제안한 전체주소생성 방식에서는 주소생성회로를 주소계산회로와 주소재배열회로로 분리하여 자체 주소생성회로의 단점인 모듈러연산을 제거하였으며, 기준 좌표 $\alpha(i, j)$와 나머지 pq개 영상점들의 주소들 사이의 주소차가 쉽게 계산되는 특징을 이용하여 효율적인 조소계산회로를 설계하였다.

각 접근형태에 대한 기준 좌표 $\alpha(i, j)$와 나머지 pq개 영상점들의 주소들 사이의 주소차는 다음과 같다.

· 2차원 블럭

$$D_B(a, b) = \begin{cases} 0, & 0 \leq a \leq p-1-i//p,\ 0 \leq b \leq q-1-j//q \\ 1, & 0 \leq a \leq p-1-i//p,\ q-j//q \leq b \leq q-1 \\ s, & p-1//p \leq a \leq p-1,\ 0 \leq b \leq q-1-j//q \\ s+1, & p-1//p \leq a \leq p-1,\ q-j//q \leq b \leq q-1 \end{cases}$$

· 수평벡터

$$D_H(b) = \begin{cases} k, & kq \leq b \leq (k+1)q-1-j//q,\ 0 \leq k \leq p-1 \\ k+1, & (k+1)q-j//q \leq b \leq (k+1)q-1,\ 0 \leq k \leq p-1 \end{cases}$$

· 수직벡터

$$D_V(a) = \begin{cases} ks, & kp \leq a \leq (k+1)p-1-j//p,\ 0 \leq k \leq q-1 \\ (k+1)s, & (k+1)p-j//p \leq a \leq (k+1)q-1,\ 0 \leq k \leq q-1 \end{cases}$$

위의 결과를 이용하여 각각의 주소차와 기준주소 $\alpha(i, j)$를 더하여 pq개의 주소를 동시에 계산할 수 있다. Park[5]에서는 행과 열 논리회로에서 기준주소 $\alpha(i, j)$와의 주소차를 계산하고, 기준주소 $\alpha(i, j)$에 이 주소차를 pq개의 덧셈기로 더하여 pq개의 주소를 동시에 구하는 방법을 제시하였다. 그러나 이 방법은 세가지 접근 형태에 대하여 행과 열 논리회로를 공통으로 사용하도록 하는데 중점을 두었기 때문에 블록과 수평벡터의 경우에는 한번의 덧셈으로 구해지는데 반하여 수직벡터의 경우에는 두번의 덧셈을 필요로하는 불일치가 생긴다. Park[6, 7]에서는 이러한 불일치를 제거하기 위하여 접근 형태의 특성에 따라 두가지 주소차 가운데 하나를 선택하도록 하는 주소차 선택 방법이 제안되었다.

이 때 계산된 pq개의 주소는 모듈정보를 가지고 있지 않기 때문에 주소 재배열회로에 의하여 m개의 모듈중 한 모듈에게로 분배해야 한다. 이 주소재배열회로는 자료 재배열회로와 같으므로 다음절에서 함께 다루기로 한다.

2) 재배열회로

일반적으로 영상행렬 내에서의 순서와 모듈 순서가 서로 다르므로 이를 일치시키는 회로가 필요하다. 즉, 모듈 위치에 근거한 물리적인 순서를 알고리듬에서 요구하는 순서로 변환하는 재배열 과정이 필요하다.

pq개 데이타 또는 주소들의 모듈 번호는 다음과 같다.

$$BL : (\mu(i+a, j+b)+aq+b)//m,\ 0\le a<p,\ 0\le b<q$$
$$HS : (\mu(i, j)+b)//m,\ 0\le b<pq$$
$$VS : (\mu(i+a, j)+aq)//m,\ 0\le a<pq$$

재배열방법에서는 많은 연구가 있었으나 간단하고 효율적인 방법으로는 Park[6, 7]이 제안한 방법을 들 수 있다. 이 방법은 다음식과 같이 pq개의 주소 또는 자료들의 기억모듈 번호들을 $\mu(i, j)$를 첫번째 모듈로하여 오름차순으로 배열시킨 다음 $\mu(i, j)$만큼 오른쪽으로 회전시키는 방법이다.

$$BL : A_2(k)\leftarrow A_1(k),\ 0\le k\le pq-1$$
$$HS : A_2(k)\leftarrow A_1(k),\ 0\le k\le pq-1$$
$$VS : A_2((kq)//m)\leftarrow A_1(k),\ 0\le k\le pq-1$$

3) 기억모듈 선택회로

보통 $m>pq$ 이므로 $m-pq$개의 모듈이 사용되지 않는다. 그러므로 전체 기억모듈 중에서 실제로 사용되는 모듈만을 선택하는 회로가 필요하다. 이러한 역할을 담당하는 회로가 기억모듈 선택회로로 m개의 모듈로부터 실제로 사용되는 다음 pq개의 기억모듈을 선택한다.

$$BL : \mu(i+a, j+b),\ 0\le a<p-1,\ 0\le b<q-1$$
$$HS : \mu(i, j+b),\ 0\le b<pq-1$$
$$VS : \mu(i+a, j),\ 0\le a<pq-1$$

실제로는 m−pq가 pq에 비해서 훨씬 작기 때문에 반대로 m−pq개의 모듈을 선택되지 않도록 하는 편이 효율적이다. 특히 m=pq+1인 경우에 선택되지 않아야되는 모듈은 다음과 같다.[6, 7]

$$BL : \mu(i+a-1, j+q)$$
$$HS : \mu(i, j+pq)$$
$$VS : \mu(i+pq, j)$$

Ⅲ. 구 현

본 장에서는 앞서의 개념적인 설계를 실제 구현하는데서 발생하는 문제점과 해결방안들에 대하여 각 부분별로 논의하기로 한다. 그리고 이를 고려한 다중접근 기억장치를 설계한다.

1. 설계 상수

설계상수의 결정에는 다음과 같은 요인들을 고려하였다.

· HDTV 수준의 해상도를 고려 하였으므로 영상행렬의 크기는 1000×1000 이상으로 한다.

· 영상처리에서 많이 사용되는 2차원 convolution 연산이 보통 3×3 블럭에 대하여 수행되므로 3×3 이상의 블럭에 대하여 동시에 접근 할 수 있도록 p, q를 3 이상으로 한다.

· p, q, s를 2의 누승으로 정하면 p, q, s로 곱하거나, 나누는 연산이 shifter에 의하여 처리되므로 회로를 단순화시킬 수 있다. 그러므로 p, q, s등의 설계상수를 2의 누승으로 한다.

· 사용되지 않는 모듈의 수를 최소로 한다.

위의 요인들을 고려하여 다중접근 기억장치의 프로토타입 제작에 사용된 설계상수는 다음과 같다.

· 기본상수 : p=4, q=4(동시 접근가능한 모듈 수는 16개)

· 영상행렬의 크기 : 1024×1024(M×N)

· 기억모듈의 수 : 17개(pq+1)

· 각 기억모듈의 수 : 17개(pq+1)

· 각 기억모듈의 크기 : 512K("전체 기억공간의 크기"/"동시에 접근가능한 기억 모듈의 수"=1M/16)

· 기타 : s=256(N/q=1024/4)

2. 구현

앞 절의 설계상수를 바탕으로 구현한 다중접근 기억장치의 블록도는 그림 5와 같다.

그림 5를 앞 장의 그림 4와 비교해 보면 pq개 영상점 중에서 일부를 선택할 수 있도록 모듈선택입력이 추가 되었고, 주소생성회로를 주소계산회로와 주소 재배열회로로 분리하였으며, 재배열회로와 기억모듈 선택회로에서 공통적으로 사용되는 모듈할당회로를 별도의 회로로 독립시켜 공유하도록 하였다.

1) 모듈할당회로

모듈할당회로는 기준좌표(i, j)의 영상점이 위치하는 모듈값을 구하는 회로로 재배열회로와 기억모듈 선택회로에서 회전회로의 제어 입력값으로 사용된다. 모듈할당함수로 $\mu(i, j)=(iq+j)//m$을 사용하였으므로 곱셈,

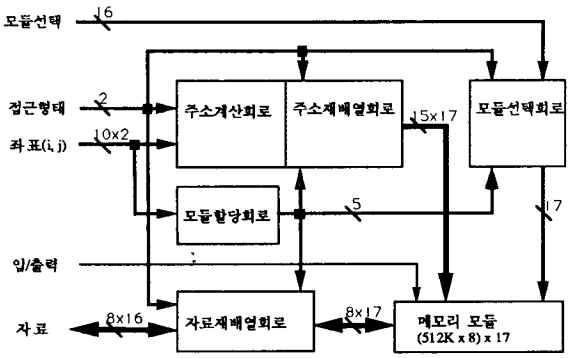


그림 5. 구현한 다중접근 기억장치의 블럭도
Fig. 5. Block diagram of multiple access memory.

덧셈, 모듈러 연산이 각각 한번씩 필요하다. 곱셈기는 q가 2의 누승인 4이므로 shift 연산으로 대치 가능하며, 실제 구현시에는 단순히 배선에 의하여 처리할 수 있다. 가산기의 경우 일반적인 직렬 가산기로는 원하는 정도의 처리속도를 얻을 수 없으므로 Carry Lookahead 가산기를 이용하여 고속화 한다. 그러나 모듈러 연산의 경우 모듈수 m이 2의 누승이 아니므로 shifter에 의하여 처리할 수 없다. 그런데 μ함수를 잘살펴보면 제수 m이 상수이므로 m으로 나눈 나머지를 구하기 위하여 일반적인 제산기 대신에 ROM으로 대치시키는 방법을 생각할 수 있다.[4] ROM의 주소공간은 Mq+N이고 워드 크기는 $\lceil \log_2 m \rceil$이다. 설계상수를 대입하여 계산하면 ROM의 크기는 5k×k비트 정도이므로 일반적인 고속 PROM으로 구현하였다.

2) 주소계산회로

주소계산회로는 기준점(i, j)의 주소를 구하기 위한 주소할당회로, 기준주소와의 주소차를 저장하는 테이블, 그리고 기준 주소와 주소차를 더하여 실제 주소를 구하는 pq개의 가산기로 구성된다. 가산기는 충분히 빠르게 설계할 수 있으므로 주소할당회로에 대해서만 고려하면 된다. 주소할당함수로 $\alpha(i, j)=\lfloor i/p \rfloor s+\lfloor j/q \rfloor$를 사용하였으므로 두번의 곱셈 연산과 한번의 덧셈 연산이 필요하다. 그런데 p, q, s가 모두 2의 누승이므로 주소할당회로의 곱셈과 나눗셈 모두 shifter만으로 처리할 수 있다. 더우기 s=N/q이므로 실제 구현시에는 모든 연산을 단순히 배선으로 처리할 수 있으므로 $\alpha(i, j)$를 계산하는 데는 별도의 시간이 불필요하다. 가산기는 주소차의 변화범위가 6비트 에 한정되기 때문에 보통의 가산기 보다 간단하게 구성할 수 있으며, 블록도를 살펴보면 계산시간이 오래 걸리는 모듈할당회로의 처리시간 내에만 주소를 계산하면 되므로 회로가 간단한 일반적인 직렬가산기로 충분하다.

3) 기억모듈선택회로

그림 4의 기억모듈선택회로는 모듈러 연산 회로가 필요한 별도의 모듈할당회로를 m−pq개 사용해야 하므로 회로가 복잡해지는 단점이 있다. 이러한 문제점을 해결하기 위해서 1비트 재배열회로를 이용하여 선택신호를 재배열하는 선택신호 재배열 방법을 이용하였다. 재배열회로의 입력값은 행우선 순서로 pq개의 비트는 '1'로 하고, 나머지 m−pq개는 '0'으로 해서 재배열회로를 거친후의 값이 '1'인 모듈만 선택되도록 하는 방법이다. 이렇게 함으로써 기억모듈선택회로만을 위한 별도의 모듈할당회로를 사용할 필요가 없다. 이러한 선택신호 재배열회로의 부가적인 장점으로는 접근집합의 일부에 대해서만 사용하고자 하는 경우 '1'로 설정한 pq개의 값을 외부에서 설정할 수 있도록 하여 쉽게 대응 할 수 있다. 그림 5의 회로에서는 이를 위하여 모듈선택입력을 추가 하였다.

4) 재배열회로

재배열회로에는 주소 재배열회로, 자료 재배열회로, 모듈선택 재배열회로가 있으며, 사용되는 워드의 크기만 다르고 각 비트마다의 회로는 모두 같다. 재배열회로는 기본적으로 Park[6, 7]의 방법을 사용하였다. 이 재배열회로는 각 접근형태마다 하나씩의 이동회로와 하나의 회전회로로 구성된다. 여서기는 회전회로에서 주로 시간이 소요되므로 이를 고속화하기 위하여 barrel shifter를 사용하였다. Park[7]은 m개의 2-입력 mux를 $\lceil \log_2 m \rceil$단계 사용한 barrel shifter를 고려하였으나, 모듈 수 m이 작은 경우에는 m^2개의 스위치를 이용한 cross bar 스위치에 의하여 1단계만에 처리하는 편이 지연시간 면에서 유리하다. 이 경우 각각의 스위치는 1개의 Transistor로 만들수 있고, 또 양방향 특성을 가지므로 자료 재배열회로의 경우 판독과 기입에 별도의 재배열회로를 들 필요가 없는 장점이 있다.

Ⅳ. 시뮬레이션 및 성능평가

시뮬레이션에는 Cadence Design System사의 로직 시뮬레이터인 Verilog-XL을 사용하였으며, SUN Sparc Station 호환의 Hyundai Workstation HWS-S100상에서 수행하였다.

그림 6은 Verilog 시뮬레이터의 wave form 출력화면을 나타낸다.

그림 6의 각 신호들에 대해서 간단히 설명하면 다음과 같다.

· Type : 접근형태, 블럭=0, 가로벡터=1, 세로벡터=2

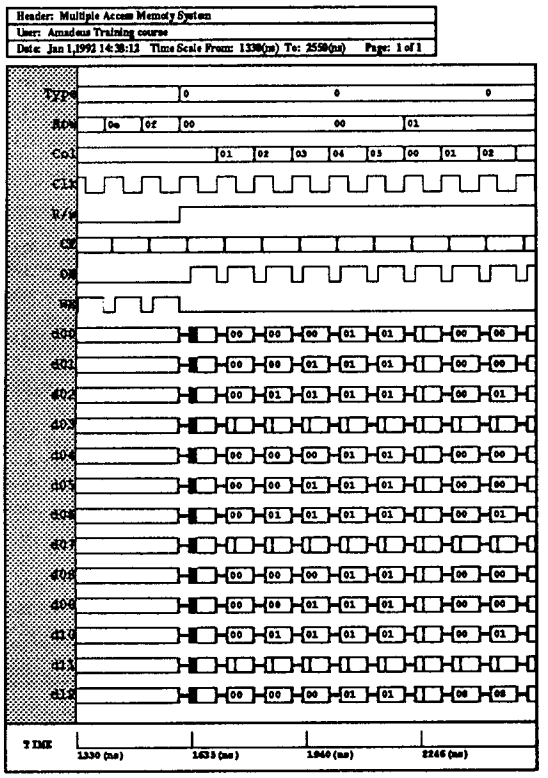


그림 6. 파형 화면
Fig. 6. Wave form screen.

· Row, Col : 기준점의 행주소와 열주소

· CE : Chip enable. 각 기억 모듈마다 하나씩 17-비트의 벡터로 표시 disable=0, enable=1

· OE, WE : 두 신호의 조합으로 연산의 종류를 결정 OE가 1이고 WE가 0이면 read 연산 OE가 0이고 WE가 1이면 write연산

· d00-d15 : 자료 입출력 버스

시뮬레이션에 사용한 기능회로의 지연 인수(delay parameter)들은 1.0-μ CMOS 공정을 이용한 ASIC으로 구현했을 경우에 대한 값이다. 지연 인수의 계산에는 다음 원칙을 사용하였다.

· 게이트 : 기본 게이트(2-입력 NAND등)는 팬 아웃이 4 이하인 경우에 대략 1(nsec)정도의 지연시간이 소요된다. 입력의 수가 더 많은 경우에는 추가 입력마다 2(nsec)를 더한다.

· 가산기 : 일반적인 직렬가산기의 지연시간은 비트당 2(nsec)이다. 모듈할당회로에 사용되는 가산기는 4비트 캐리 계산회로를 이용하여 속도를 가속화시켜 사용한다. 4비트 캐리 계산회로의 지연시간은 4(nsec)정도이므로 n-비트 가산기의 전체 지연시간은 $\lceil n/4 \rceil \times 4$(nsec)가 된다.

· 테이블 : 테이블의 크기가 작은 경우에는 지연 시간이 주로 주소디코더의 입력수에 비례하므로 ($\lceil \log_2 n \rceil$*0.2+2)(nsec)가 된다. 그러나 테이블의 크기가 커지면 주소디코더의 게이트 수가 급격히 늘어나서 디코더를 구동하기 위한 드라이버가 필요하게 되므로 지연시간은 큰폭으로 늘어나게 된다. 그러므로 모듈할당회로의 모듈러연산용 테이블의 지연 인수는 고속 PROM의 접근시간을 이용하였다.

· 출력포트 : 출력 버퍼의 지연시간은 대략 3(nsec)이다.

정확한 지연시간을 계산하기 위해서는 각 게이트의 팬아웃과 배치후의 배선길이 등의 요소들에 대해서도 고려해야하나, Verilog의 지연 모형으로 나타내기가 어려워서 팬아웃이 크다고 여겨지는 부분에는 지연시간이 2(nsec)인 버퍼를 두는 것으로 계산하였다.

위의 원칙을 사용해서 각 회로별로 지연시간을 계산하면 다음과 같다.

· 주소계산회로(t_{AC})

주소계산회로의 지연시간은 주소차를 위한 테이블의 지연시간과 기준주소 $\alpha(i, j)$와 주소차를 더하기 위한 가산기의 지연시간의 합으로 구성된다.

−테이블 참조 : 10(nsec)(4(2−bit)+6(3bit))

−16-6비트 가산기 : 22(nsec)(2*16−10)

−주소 계산회로의 지연시간 : 32(nsec)

· 모듈할당회로(t_{MA})

모듈할당회로의 지연시간은 가산기와 모듈러 연산을 위한 테이블의 지연시간의 합으로 구성된다. 모듈러 연산을 위한 테이블의 경우 주소공간이 크기때문에 고속 PROM의 자료를 사용하였다.

−12-비트 가산기 : 12(nsec)(4*3)

−모듈러 연산(테이블 참조) : 22(nsec)(20(PROM의 접근시간)+2(driver))

−모듈할당회로의 지연시간 : 34(nsec)

· 자료 & 주소 재배열 회로(t_{DR}, t_{AR})

재배열회로의 지연시간은 mover, barrel shifter 그리고 메모리 모듈을 구동하기 위한 출력 버퍼의 지연시간의 합으로 구성된다.

−Mover : 1(nsec)

−Barrel Shifter : 2(nsec)

−Output buffer : 3(nsec)

−재배열 회로의 지연시간 : 6(nsec)

· 메모리 모듈 접근시간(t_{MA}, t_{MW})

메모리 모듈은 일반적인 고속 CMOS SRAM을 고려하였다. 고속 CMOS SRAM의 접근시간은 대략 25-30(nsec)이다.

각 기능 회로별 지연시간을 합해서 전체 기억장치의 지연시간을 계산하면 다음과 같다.

· Read 접근시간

$t_R = \max\{t_{AC}, t_{MA}\} + t_{AR} + t_{MR} + t_{DR}$

$= \max\{32, 34\} + 6 + 30 + 6$

$= 76(\text{nsec})$

· Write 접근시간

$t_W = \max\{t_{AC}, t_{MA}\} + \max\{t_{AR} + t_{DR}\} + t_{MW}$

$= \max\{32, 34\} + 6 + 30$

$= 70(\text{nsce})$

2차원 영상 1 프레임의 모든 영상점에 한번씩 접근하는데 소요되는 시간은 "1 프레임의 크기"×"접근시간"/"동시에 접근되는 영상점의 수"이므로, 접근시간을 80(nsec)로 하고 계산해보면 1024×1024×80(nsec)/16≃50(msec)이 된다. 1개의 모듈을 사용한 기억장치의 경우는 1024×1024×30(nsec)≃300(msec)이므로 6배 정도 성능이 향상 되었음을 알 수 있다.

V. 결 론

대량의 영상자료를 고속으로 처리하기 위해서 영상점에 대한 접근시간을 줄이기 위한 방법의 하나로 여러형태의 영상점 집합에 대하여 동시접근 가능한 다중접근 기억구조가 많이 연구되었다. 본 논문에서는 이러한 다중접근 기억구조의 전체 시스템 측면에서의 성능향상과 회로의 단순화에 중점을 두었다. 이를 위하여 기억모듈 선택회로에 재배열회로를 확장 적용하여 별도의 모듈러 연산을 사용하지 않도록 하였으며, 속도와 chip 면적을 고려하여 계산경로간의 지연시간차를 줄임으로써 회로를 최적화하였다. 개선된 다중접근 기억장치는 Verilog-XL로 시뮬레이션하여 올바르게 동작함을 검증하고 성능을 평가하였다.

설계한 다중접근기억장치는 4×4의 2차원 블록, 1×16의 수평벡터, 그리고 16×1의 수직벡터에 대하여 동시접근이 가능하며, 보통의 기억장치에 비하여 6배 이상의 처리속도를 갖는 것으로 분석되었다.

앞으로의 연구 방향은 설계한 다중접근 기억장치를 효과적으로 사용할 수 있는 프로세서를 개발하고, 기존의 영상처리 알고리듬을 여기에 적합하도록 개선하는 것이다.

參 考 文 獻

[1] P. Budnik and D. J. Kuck, "The organization and use of parallel memories," *IEEE Trans. comput.*, vol. C-20, no. 12, pp.1566-1569, Dec. 1971.

[2] D. H. Lawrie, "Access and alignment of data in an array processor," *IEEE Trans. Comput.*, vol. C-24, no. 12, pp. 1145-1155, Dec. 1975.

[3] D. C. Van voorhis and T. H. Morrin, "Memory System for Image Processing," *IEEE Trans. Comput.*, vol. C-27, no. 1, pp. 113-125, Feb. 1978.

[4] D. H. Lawrie and C. R. Vora, "The prime memory system for image processing," *IEEE Trans. Comput.*, vol. C-31, no. 5, pp. 435-442, May 1982.

[5] J. W. Park, "An efficient memory system for image processing," *IEEE Trans. Comput.*, vol. C-35, no. 7, pp. 669-674, Jul. 1986.

[6] J. W. Park, "Efficientimage analysis memory system," *Journal of The Korea Information Science Society*, vol. 16, no. 6, pp. 559-566, Nov. 1989.

[7] J. W. Park, *Multi-Access Memory System for Image Processing*, Ph. D. Dissertation C. S. Dept. KAIST, 1991.

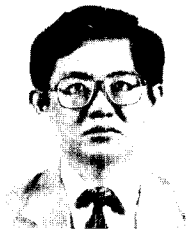
[8] 박종원 외, "영상 미디어의 Multi-resolution 표현 변환 기법과 피라미드 구조 알고리듬 설계에 관한 연구," 최종연구보고서, 한국전자통신연구소, 1992.

著 者 紹 介

金 洁 潤 (準會員)

1967年 5月 15日生. 1990年 2月 한양대학교 공과대학 산업공학과 졸업(공학사). 1991年 3月～현재 한국과학기술원 전산학과 석사과정. 주관심분야는 병렬처리, 영상처리, VLSI설계 등임.

●



李 興 揆 (正會員)

1955年 10月 3日生. 1978年 2月 서울대학교 공과대학 전자공학과 졸업(공학사). 1981年 2月 한국과학원 전산학과 졸업(이학석사). 1984年 8月 한국과학기술원 전산학과 졸업(공학박사). 1986年 8月 미국 미시간 대학 Manufacturing Center 연구원. 1986年 9月～현재 한국과학기술원 조교. 1987年 8月 영국 Imperial College in London 교환교수. 주관심분야는 병렬처리, VLSI설계, 실시간 및 고장허용 시스템 등임.

朴 宗 元 (正會員)

1979年 2月 충남대학교 공과대학 전자공학과 졸업(공학사). 1981年 2月 한국과학원 전산학과 졸업(이학석사). 1991年 8月 한국과학기술원 전산학과 졸업(공학석사). 현재 충남대학교 전산학과 부교수. 주관심분야는 컴퓨터 구조 및 영상 분석 등임.