논문 00-02-04

영상처리를 위한 Pipelined 병렬처리 시스템

Pipelined Parallel Processing System for Image Processing

李 炯*, 金 重 培**, 崔 成 赫***, 朴 宗 元****

(Hyung Lee*, Jong Bae Kim**, Sung Hyk Choi***, Jong Won Park****)

요 약

본 논문에서는 영상 응용프로그램의 처리 속도를 향상하기 위한 병렬처리 시스템을 제안한다. 병렬처리 시스템은 Pipelined SIMD 구조를 갖고 있으며, 다수개의 처리기와 다중접근 기억장치로 구성된다. 다중접근 기억장치는 메모리 모듈들과 메모리 제어부로 구성되며, 메모리 제어부는 메모리 모듈 선택 모듈, 데이터 라우팅 모듈, 그리고 주소 계산 및 라우팅 모듈로 구성되어 있으며, 블록, 행, 그리고 열 내의 데이터를 동시에 접근할 수 있는 기능을 제공한다. 제안한 병렬처리 시스템을 검증하기 위해서 형태학적 필터를 적용하여 기능 검증 및 처리 속도를 확인하였다.

Abstract

In this paper, a parallel processing system is proposed for improving the processing speed of image related applications. The proposed parallel processing system is fully synchronous SIMD computer with pipelined architecture and consists of processing elements and a multi-access memory system. The multi-access memory system is made up of memory modules and a memory controller, which consists of memory module selection module, data routing module, and address calculating and routing module, to perform parallel memory accesses with the variety of types: block, horizontal, and vertical access way. Morphological filter had been applied to verify the parallel processing system and resulted in faithful processing speed.

Keywords : Parallel Processing System, Multi-access Memory System; Image Processing, SIMD Architecture

* 忠南大學校 컴퓨터工學科
(Department of Computer Engineering, Chungnam National University)

** 욱성電子
(WOOKSUNG Electornics, Inc.Dept. of .Elec. Eng., KonKuk Univ.)

接受日: 2000年8月25日 修正完了日: 2000年11月18日

***忠南大學校 컴퓨터工學科
(Electronics and Telecommunications Research Institute(ETRI))

****忠南大學校 情報通信工學科
(Department of Information Communications Engineering, Chungnam National University)

I. 서 론

영상, 비디오, 그리고 그래픽과 같은 시각적 매체들을 실시간으로 처리하기 위한 구현 기술과 그에 따른 확장성 측면에서 많은 연구들이 진행되고 있다. 이러한 연구들은 멀티미디어를 실시간으로 처리하기 위하여 특정 매체를 위한 특정 프로세서 구현부터 여러 매체들을 함께 처리할 수 있는 프로세서 구현을 포함하는 범주까지 진행되고 있다. 또한, 다양한 병렬처리 기법들이 시각적 매체들을 실시간으로 처리하기 위한 프로세서 구현에 적용되고 있다. 시각적 매체들의 경우, 다른 매체들과는 달리 방대한 정보량과 정보에 따른 많은 데이터를 갖고 있다. 이러한 시각적 매체를 실시간으로 처리할 수 있는 하드웨어 구현을 위하여 복잡성, 다양한 기술과 도구들, 빠른 처리 속도, 그리고 저장장치 및 입출력 대역폭 등에 관련된 많은 연구가 이루어지고 있지만, 독립적인 부분 연구로 진행되고 있기 때문에 전체 시스템 차원에서 본다면 서로간의 연관성을 깊이 고려해야 한다.

현재 연구 중인 컴퓨터 시스템에서는 메모리 대역폭을 높이기 위하여 다수의 메모리 모듈을 사용한 인터리빙(interleaving) 방식이나 병렬접근 메모리 시스템을 고려하고 있다. 이러한 메모리 시스템에서 효율적인 대역폭을 지원하기 위해서는 많은 메모리 모듈과 고속 접근이 가능한 메모리 모듈을 사용해야 하지만, 다수개의 처리기들이 동시에 같은 메모리 모듈에 접근을 시도함으로써 발생할 수 있는 메모리 접근 충돌(conflict)을 고려하여야 한다. 특히, 영상처리 알고리즘들은 접근될 영상 데이터의 배열 형태가 블록, 행, 열, 대각선, 그리고 역대각선인 경우에 대하여 빈번하게 반복적인 연산을 수행하기 때문에 다양한 형태 내의 다수개의 영상 데이터를 신속하게 접근 할 수 있는 병렬접근 메모리 시스템과 이에 따른 처리기의 개발이 요구되고 있다.

일반적으로, 실제적인 연산을 수행하는 다수개의 처리기(Processing Element)와 처리기들을 제어하기 위한 제어장치, 명령어들을 저장하기 위한 공유메모리, 다수개의 데이터를 동시에 접근 가능하도록 하는 병렬접근 메모리 시스템들로 구성된 SIMD (Single-Instruction Multiple-Data stream) 처리기 구조를 갖는 병렬처리 시스템에 관한 연구가 진행되고 있는데, 이러한 구조를 갖는 시스템이 다양한 영상 처리를 위한 응용 시스템으로 고려되고 있다[1]-[7]. SIMD 구조를 갖는 병렬처리 시스템은 병렬접근 메모리 시스템으로부터 다수개의 데이터에 접근하기 위해서 제어장치가 공유 메모리 모듈로부터 명령어를 읽은 후, 다수개의 처리기에 접근 방식, 기준 좌표, 그리고 접근할 데이터들의 간격을 전송한다. 그리고, 처리기들이 서로 다른 데이터들에 대한 명령을 수행하기 위하여 제어장치가 다수개의 처리기에 동일한 명령어를 전송한다. 이렇게 함으로써, 다수개의 처리기가 서로 다른 데이터에 대해 동일한 명령을 수행하게 되고, 다수개의 서로 다른 데이터를 동시에 접근할 수 있다.

SIMD 처리기 구조를 갖는 시스템의 경우, 처리기의 효율성을 극대화하기 위해서 병렬접근 메모리 시스템이 다음과 같은 특성을 만족해야 한다. 1) 병렬접근 메모리 시스템은 다양한 접근 형태(블록, 행, 열, 대각선, 그리고 역대각선 등)내의 데이터를 동시에 접근할 수 있어야 한다. 2) 메모리 내의 임의의 위치에 있는 모든 데이터들은 주어진 접근 형태에 따른 동시 접근이 가능해야 한다. 3) 데이터 라우팅과 주소 계산 및 라우팅 회로가 단순하고 처리 속도가 빨라야 한다. 4) 데이터 라우팅과 주소 계산 및 라우팅을 위한 처리 루틴이 처리기와 독립적으로 수행되어야 한다. 5) 병렬접근 메모리 시스템을 구성하는 메모리 모듈의 개수가 처리기의 개수보다 같거나 많지만 가능한 적어야 한다.

본 논문에서는 이러한 특성을 만족하는 병렬접근 메모리 시스템으로는 다중접근 기억장치를, 입출력은 PCI 버스를, 그리고 Pipelined SIMD 처리기 구조를 갖는 전체 시스템 차원의 영상 처리용 병렬처리 시스템을 제안한다. 제안한 병렬처리 시스템은 ALTERA사의 FPGA (Field Programmable Gate Array)에 fitting 하였으며, 다중접근 기억장치와 연계된 처리기의 개수는 8개로 하였다. 그리고 호스트 컴퓨터와의 인터페이스는 PCI 버스를 채택하였으며, 처리기들의 제어를 위한 제어장치는 구현의 용이성과 범용성, 유연성 등을 고려하여 인텔사의 i960 프로세서를 사용하였다. 그리고,

CADENCE사의 Verilog-XL을 이용하여 병렬처리 시스템에 대한 모의 실험을 수행하여 기능 및 성능을 검증하였고, 이를 i960RM과 제안한 병렬 시스템의 이전 버전과 비교하여 실제 얻어진 성능 향상의 효과를 확인하였다.

본 논문의 구성은 제 II장에서 관련 연구 및 동향을 나열하고, 제 III장에서는 병렬메모리 시스템인 다중접근 기억장치에 대해서 기술한다. 그리고, 영상처리를 위한 pipelined 병렬처리 시스템의 세부 사항에 대해서는 제 IV장에서 기술하고, 제안한 시스템의 성능측정 및 분석은 제 V장에서 기술한다.

II. 관련 연구 및 동향

일반적인 영상처리 및 특정 영상처리 알고리즘을 수행 할 수 있는 다양한 컴퓨터 시스템들에 대한 많은 연구와 결과들이 제시되었는데, 일반적으로 이들 시스템들은 크게 두 가지 분류로 나눌 수 가 있다. 첫 번째는 기본적인 전체 영상을 수행하거나, 한번에 병렬처리가 가능한 부분적인 영상을 처리할 수 있도록 프로세서들의 2차원 배열 구조로 구성된 시스템이다. 이런 종류의 시스템들은 CLIP[4], MPP[5] 등이 있으며, 일반적으로 이들은 SIMD 구조를 갖고 있다. 이들 시스템들의 주요 단점은 비용이 많이 든다는 것이다. 예를 들면 512 x 512 영상을 처리하기 위해서는 대략 25만개의 처리기(Processing Element)가 필요하게 된다. 더욱이 입력 영상의 직렬적인 특성 때문에 처리기들의 최대 성능을 얻을 수가 없다. 두 번째 분류는 특정 크기의 윈도우 내에서 영상을 처리하는 지역 윈도우 처리기 시스템이다. MITE[6], PIPE[7] 등이 이런 분류에 속하는데, 이들은 처리할 영상의 크기가 증가할수록 일정한 처리 속도를 유지하기 위해서 처리기 속도의 기하학적이 증가를 요구하게 된다. 이런 종류의 시스템들은 프로그래밍이 가능하기 때문에 융통성이 있어 다양한 응용에 적용될 수 있지만, 상대적으로 속도가 느리며, 특히, 특정 영상처리를 위한 구현에는 가격 효율성 측면에서 적합하지 않다는 단점을 갖고 있다. 이상의 시스템들은 영상 처리를 수행하는 처리기 관점에서 구현된 시스템들이며, 영상은 방대한 정보량과 정보에 따른 많은 데이터를 갖고 있기 때문에 데이터를 효율적으로 처리하기 위한 메모리 시스템에 대한 연구도 진행 중이다.

영상 처리와 관련된 병렬접근 메모리 시스템은 자료의 블록, 행, 열, 대각선, 그리고 역대각선 등을 동시에 접근하는데 있어서 메모리 충돌을 방지하기 위한 많은 연구가 진행되어 왔다. 1971년에 P. Budnik과 D. J. Kuck [8]은 Linear Skewing Scheme과 Prime Memory System을 소개하였는데, 이것은 $N \times N$으로 구성된 데이터에서 메모리 모듈의 개수를 N보다 큰 소수 이상으로 구성하면 행, 열, 대각선, 역대각선, $N^{1/2} \times N^{1/2}$블록을 메모리 충돌 없이 동시에 접근할 수 있다고 하였다. 그러나, 이 방법은 주소 계산을 위하여 소수의 modulo 연산을 요구하여 회로가 복잡하고 항상 일정한 시간에 수행하기가 어렵다는 단점을 지니고 있다. D. H. Lawrie [9]는 1975년에 Linear Skewing Scheme을 일반화시키고, 주소생성 회로를 단순화한 Array Storage Scheme을 제안하였다. 이 방법은 $2N$ 메모리 모듈로 구성되며 행, 열, 대각선, 역대각선, $N^{1/2} \times N^{1/2}$블록 내의 데이터를 동시에 접근할 수 있다고 하였다. 그러나, 이 방법은 메모리 모듈의 낭비가 크다는 것이 단점으로 지적되고 있다. 이와 같이 주소생성 회로가 복잡하고 메모리 낭비가 크다는 Linear Skewing Scheme의 단점을 보완하기 위하여 Lee [10]는 행, 열, 정방형 블록 및 분산된 블록 내의 데이터를 동시에 접근할 수 있는 Nonlinear Skewing Scheme을 제안하였는데, 이것은 주소 계산을 위해 요구되었던 modulo 연산을 제거한 것이다. 그러나, Nonlinear Skewing Scheme은 대각선 접근에 대해서는 효율적이지 못한 단점이 있다.

Van Voorish와 Morrin [11]은 $pq+1$개의 메모리 모듈로 구성되고 블록($p \times q$), 행($1 \times pq$), 열($pq \times 1$) 내의 데이터를 동시에 접근할 수 있는 기억장치를, Lawrie와 Vora [12]는 블록에 대한 설계가 제외된 행, 열, 대각선, 역대각선에 대한 병렬접근 메모리 시스템을 제안하였다. 그러나, Van Voorish와 Morrin은 2의

누승이 아닌 임의의 숫자에 대한 modulo 연산이 필요하고, Lawrie와 Vora는 블록 접근이 제외되어 있으므로 블록, 행, 열에의 접근이 신속하고 빈번하게 이루어져야 하는 영상처리 및 통신분야에서는 이러한 메모리 시스템의 적용이 어렵다는 단점을 지니고 있다. 그리고, Raghavendra 등[13]이 발표한 메모리 시스템은 접근하는 자료들이 2차원 배열 내의 특정한 위치에 있어야 하기 때문에 임의의 위치에 있는 자료들을 신속하게 접근해야 하는 영상처리 및 통신 분야에 유용하게 적용될 수 없다.

Park의 메모리 시스템 [14]은 Van Voorish와 Morrin의 메모리 시스템을 개선하여 주소생성 회로를 단순화 시켰다. 1986년에 발표된 메모리 시스템을 확장하여 1989년에는 대각선과 역대각선 내의 데이터를 접근할 수 있는 메모리 시스템을 개발하였고, 임의의 각도를 갖는 선분들을 신속하게 접근할 수 있는 메모리 시스템과 해상도를 어느 정도 낮추어 적은 영상 데이터 내에서 신속하게 움직임 탐지나 Texture 분석을 할 수 있는 가우시안 피라미드 형성에 적합한 간격이 2인 행을 접근할 수 있는 메모리 시스템을 제안하였다[15].

본 논문에서 제안한 병렬처리 시스템은 기본적으로 SIMD 구조를 가지고 있으며, 주소생성 회로가 단순하며 영상처리에 필수적인 블록, 행, 열 내의 데이터를 동시에 접근 할 수 있는 Park의 병렬접근 메모리 시스템인 다중접근 기억장치를 적용함으로써 앞서 언급한 두 분류의 시스템들이 갖는 단점을 해결하였다.

III. 다중접근 기억장치

디지털 영상을 화소들의 집합, 즉, $I(*,*)$의 $M\times N$ 행렬로 나타낼 수 있는데, 각 화소 $I(i,j)$는 $0\le i\le M-1$과 $0\le j\le N-1$에 대한 해당 부분의 색과 밝기를 나타낸다. 임의의 정수 p, q를 설계 상수로 했을 때, pq개의 화소들은 블록($p\times q$), 행($1\times pq$), 그리고 열($pq\times 1$)의 세 가지 형태로 접근될 수 있으며 $I(*,*)$에서 각각의 접근 형태에 따라 동시 접근되는 블록, 행, 그리고 열 내의 데이터는 다음과 같이 나타낼 수 있다.

$$BL(i,j) = \left\{ \begin{array}{l} \{I(i+a,j+b) \mid 0\le a\le p,\ 0\le b\le q\}, \\ 0\le i\le M-p,\ 0\le j\le N-q \end{array} \right\}$$

블록,

$$HS(i,j) = \left\{ \begin{array}{l} \{I(i,j+b) \mid 0\le b\le pq\}, \\ 0\le i\le M,\ 0\le j\le N-pq \end{array} \right\}$$ 행,

$$VS(i,j) = \left\{ \begin{array}{l} \{I(i+a,j) \mid 0\le a\le pq\}, \\ 0\le i\le M-pq,\ 0\le j\le N \end{array} \right\}$$ 열.

이러한 여러 가지 접근 형태에 대하여 동시에 접근이 가능하도록 메모리 모듈을 배치하는 모듈할당 함수와 배치된 모듈 내의 주소를 결정하는 주소할당 함수가 요구되며, 다중접근 기억장치는 이러한 기능함수들로 구성된다.

3.1 모듈할당 함수(μ)

동시에 접근되는 데이터의 집합 S를 동시에 읽거나 쓰기 위해서는 S의 각 원소들이 서로 다른 메모리 모듈에 있어야 한다. 메모리 모듈의 수를 m이라 한다면 S의 각 원소들은 서로 다른 m개의 메모리 모듈 내에 위치해야 한다. 어떤 데이터가 어떤 모듈에 위치할 것이지를 결정하는 함수 μ를 모듈할당 함수(module assignment function)라 하고 다음과 같이 정의한다.

$$\mu : I(i,j) \Rightarrow c(0\le c\le m)\ ,0\le i\le M,\ 0\le j\le N$$

그리고, 집합 S의 각 원소를 동시에 접근하기 위해서 모듈할당 함수 μ는

$$(i1,j1),(i2,j2) \in S \Rightarrow \mu(i1,j1) \neq \mu(i2,j2)$$

와 같은 특성을 갖는다.

동시에 접근되는 메모리 모듈은 pq개이므로 m이 pq보다 큰 경우에는 $m-pq$개의 메모리 모듈은 사용되지 않는다. 따라서, $m=pq$인 경우, 메모리의 이용 측면에서는 가장 효율적이지만 동시에 접근 가능한 형태는 블록, 행, 열의 일부분만 가능하며, 이 세 가지 형태 모두에 대해서 동시에 접근이 가능한 함수 μ는 존재하지 않기 때문에[7], m은 pq보다 적어도

하나가 많아야 한다. μ는 보통 skew 함수[4]-[8]를 사용하여 계산하는 경우가 많다.

3.2 주소할당 함수(α)

동시에 접근되는 데이터의 집합 S의 원소들에 대응하는 메모리 모듈은 함수 μ에 의해서 결정되지만, 데이터가 위치하는 메모리 모듈 내의 주소도 동시에 함께 계산 되어져야 한다. 따라서, S의 각 원소에 대하여 함수 μ에 위해 결정된 모듈 내의 주소를 계산하는 함수 α가 필요하다. 이를 주소할당 함수(address assignment function)라 하고 다음과 같이 정의한다.

$\alpha : I(i,j) \Rightarrow K, \ (0 \le k \le MC)$, 여기서 MC는 메모리 용량임.

그리고, 함수 α는 함수 μ와 함께 다음 조건을 만족해야 한다.

$$(i1,j1) \neq (i2,j2) \text{ and } \mu(i1,j1) = \mu(i2,j2) \Rightarrow \alpha(i1,j1) \neq \alpha(i2,j2)$$

함수 α에 대해서는 많은 연구들이 진행되어 왔으나, 그 중 한가지를 살펴보면 다음과 같다:

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

여기서 s는 다음 조건을 만족하는 임의의 정수이다:

$$s \ge \lceil N/q \rceil , \ (s \times N/p \le MC).$$

함수 α를 통해서 집합 S의 원소들에 대한 서로 다른 메모리 모듈 내의 주소들은 함수 μ에 독립적으로 임의의 기준위치 (i,j)에 의해 결정됨을 알 수 있다. 즉, 함수 α와 함수 μ는 서로 독립적이며, 두 함수의 조합으로 접근되는 데이터의 위치 지정을 위한 메모리 모듈과 주소를 유일하게 지정할 수 있다.

3.3 다중접근 기억장치의 기능회로

다중접근 기억장치의 기능블록은 메모리 내의 데이터에 대한 주소를 생성하는 주소생성 회로, 영상 행렬 내의 위치와 실제 메모리 모듈의 위치를 일치시키기 위한 데이터 및 주소 재배열 회로, 실제 사용되는 메모리 모듈을 선택하는 메모리 모듈 선택회로, 그리고 데이터가 저장되는 메모리 모듈로 구분된다.

주소생성 회로는 모듈할당 함수 μ와 주소할당 함수 α를 사용하여 이루어지는데, 기준좌표 $I(i,j)$와 접근 형태가 결정되면 각 메모리 모듈에서는 접근 집합 S의 한 원소를 갖게 되며, 주소생성 회로는 그 원소가 위치하는 메모리의 주소를 생성하게 된다. Park [10]이 제안한 전체 주소생성 방식에서는 주소생성 회로를 주소계산 회로와 주소재배열 회로로 분리하여 modulo 연산을 제거함으로써 기존 주소생성 회로의 단점을 보완하였으며, 기준 좌표의 주소 $\alpha(i,j)$와 나머지 pq개이 화소들의 주소차가 쉽게 계산되는 특징을 이용하여 효율적인 주소계산 회로를 제시하였다.

IV. 병렬처리 시스템

Park의 다중접근 기억장치를 이용한 병렬처리 시스템은 인쇄체 문자 인식[16], 얼굴 영상 검색[17], 문화재 검색[18] 등 여러 응용 분야에 연구되어 적용되었으나 시스템 구조 및 구현상의 문제점으로 인하여 이론적으로 검증된 성능이 실제 측정된 성능에 미치는 못하는 결과를 가져왔다. 또한, 적용한 알고리즘의 특정 부분이 병렬화가 어려울 때 이를 호스트 컴퓨터가 수행하게 함으로써 빈번한 데이터 전송이 발생하고, 이는 곧 성능 저하의 원인이 되었다. 본 논문에서는 이러한 시스템 구조상의 문제점으로 인하여 발생하는 성능 저하를 줄이기 위한 시스템 구조를 제안하며, 다중접근 기억장치의 구현을 간략화하여 회로의 복잡도를 줄임으로써 메모리 접근속도 및 집적도를 높이도록 하였다. 또한, 처리기의 명령어 인출(instruction fetch), 메모리 접근 및 명령어 수행을 pipelined 형태의 구조를 취함으로써 이전의 시스템 [16]-[18]에서 발생된 성능상의 문제점을 개선하였다.

제안된 병렬처리 시스템의 블록도는 그림 1와 같으며, 호스트 컴퓨터와의 정합 장치를 PCI 버스로 구성하여 PCI 버스를 지원하는 일반 컴퓨터에 적용이 가능하도록 하였다. 프로세서는 PCI 정합 장치와 메모리 제어기를 내장하고 있는 인텔사의 I80960VH를 사용하였으며, 호스트 컴퓨터와의 통신은 PCI 버스의

mailbox를 통한 메시지 형태와 인터럽트를 사용하였다. 프로세서는 호스트 컴퓨터로부터 메시지를 받으면 메시지의 내용을 해석하여 처리기가 수행할 명령어들을 명령어 큐(instruction queue)에 저장하고, 주기억 장치에 있는 데이터를 다중접근 기억장치로 전송한다. 그리고, 제어장치가 명령어 큐에 있는 명령어들을 인출하여 처리기 모듈로 전달하게 한다. 처리기 모듈은 전달된 명령어를 해석하고 그에 따른 메모리 접근이나 산술·논리연산, 조건 검색 등을 수행한다. 모든 명령어 수행이 완료되면 그 결과는 다중접근 기억장치에 저장되고, 프로세서는 처리된 데이터를 주기억 장치로 전송하고 호스트 컴퓨터로 인터럽트를 발생시켜 수행이 완료 되었음을 알린다. 처리기 모듈과 제어장치 및 프로세서의 로컬 버스는 동일한 클럭에 의해서 동기화 된 태로 동작한다

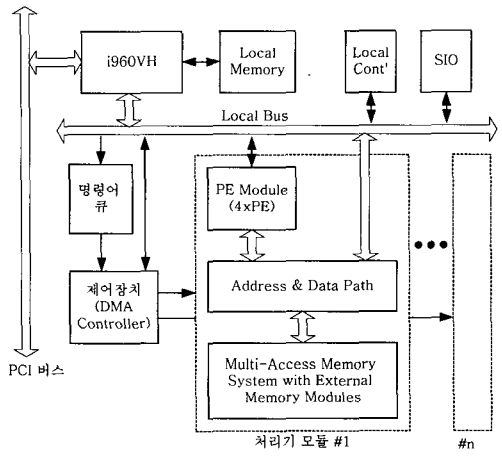


그림 1. 제안한 병렬처리 시스템의 블록도

Fig 1. Block diagram of the parallel processing system we proposed

제안한 병렬처리 시스템 내의 다중접근 기억장치와 연계한 처리기 모듈은 FPGA의 집적도를 고려하여 4개의 처리기를 가지도록 하였으며, 적용할 응용 프로그램의 병렬성, 요구되는 속도 등에 따라 추가될 수 있다. 이 모듈은 4개의 처리기와 처리기들이 수행할 명령어들을 해석하고 처리기들간의 동기화를 유지하기 위한 상태 천이기, 주소 계산 및 자료 재배열 회로, 그리고, 메모리 모듈 및 멀티 플렉서, 버퍼 등으로 구성되어 있다. 프로세서와 제어장치가 동시에 접근할 경우에 발생할 수 있는 메모리 충돌을 방지하기 위해 버스 중재기를 명령어 큐에 두었다. 처리기 모듈의 상세 블록도는 그림 2와 같다.

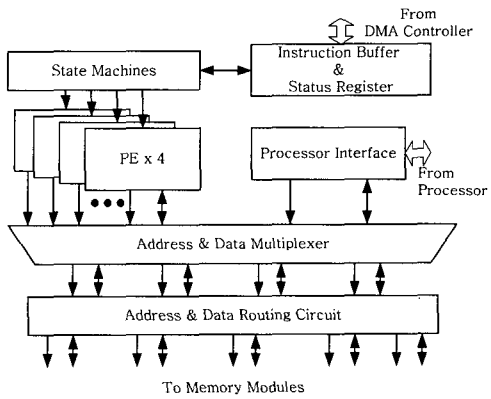


그림 2. 처리기 모듈의 블록도

Fig 2. Block diagram of Processing module

4.1 설계

제안한 병렬처리 시스템은 일반적인 영상 처리 알고리즘에 쉽게 적용될 수 있도록 확장성과 범용성을 갖도록 설계되었다. 기존의 SIMD 처리기에서는 각 처리기에서 수행되는 인스트럭션과 동기화 작업을 제어장치가 제어하였으나, 이러한 역할을 수행하기 위한 제어장치의 기능이 다양하지 않아서 다양한 병렬 응용 프로그램을 적용할 경우는 많은 제약이 뒤따른다. 반면에, 제어 장치를 범용 마이크로 프로세서를 사용하여 구현할 경우, 응용 프로그램의 개발이 용이하고 시스템의 융통성과 효율성이 증가할 수 있으나, 이 경우에는 프로세서와 처리기들간의 동기화에 따른 지연시간이 많이 발생한다. 따라서, 프로세서를 사용하여 시스템의 효율성을 높이면서도 동기화에 따른 overhead를 줄이기 위하여 기능이 축소된 제어 장치와 명령어 큐를 설계하였다.

제안한 병렬처리 시스템은 다양한 응용 프로그램에 적용하기 위해서 각 응용 프로그램에 따라 선택적으로 처리기의 개수와 메모리 모듈의 크기를 가변할 수 있도록, 즉, 처리기의 개수와 메모리 모듈의 크기 및 개수를 운용 방식에 따라 선택적으로 조절 가능하도록 4개의 처리기로 처리기 모듈을 구성하고, 처리기

모듈 단위로 증설이 가능하도록 하였다.

처리기 모듈을 설계함에 있어서 명령어 인출, 명령어 수행 및 메모리 접근을 pipelined 방식으로 설계하였고, 메모리 접근 명령과 일반 명령어를 구분하여 동시에 인출하고 수행하게 함으로써 최적의 성능을 유지할 수 있도록 하였다.

4.2 명령어 집합

처리기 모듈에서 지원하는 명령어 집합은 기존의 인쇄체 문자인식, 얼굴 및 문화재 검색, 그리고 3D 그래픽 가속기 등에서 사용된 명령어들을 조사하여 결정하였으나, 사용한 FPGA 디바이스의 집적도를 고려하여 부동소수점 연산은 제외하였다. 명령어는 메모리 접근을 위한 것과 기본적인 산술·논리 연산 및 조건 검색 등을 지원하고 있으며, 곱셈기를 제외한 모든 일반 명령어들은 1 클럭의 수행주기를 갖는다.

4.3 프로세서 모듈

프로세서 모듈은 호스트 컴퓨터로부터 처리기가 수행할 명령, 즉, 기준 좌표 (i, j), 접근 유형 (BL, VS, HS), 간격, 판독 명령, 기록 명령 등을 전달받아 명령어 큐에 저장하고, 처리기 모듈을 제어하여 명령어를 수행하게 하며, 필요에 따라 프로세서가 직접 일부 명령어들을 수행하기도 한다. 또한, PCI 버스를 통하여 호스트 컴퓨터로부터 데이터를 가져오거나 수행한 결과를 전송한다. 프로세서는 프로그램 수행을 완료하였거나 수행 도중에 에러가 발생하면 PCI 버스의 인터럽트를 통하여 호스트 컴퓨터로 알려준다. 프로세서는 PCI 브리지와 메모리 제어기를 내장한 인텔사의 I80960VH를 사용하였다.

로컬 메모리는 128KB의 Flash ROM과 4~16 MB의 DRAM으로 구성되어 있으며, 시스템 부팅 및 프로세서가 수행할 코드를 저장한다. 그밖에 주소 디코딩 회로, 인터럽트 발생 회로 및 디버깅을 위한 RS-232 인터페이스 등으로 구성되어 있다.

4.4 명령어 큐와 제어장치

제어장치는 명령어 큐에 저장된 명령어들을 순차적으로 읽어서 처리기 모듈로 전달하는 역할을 한다. 프로세서는 명령어 큐에 처리기들이 수행할 명령어들을 저장한 후 제어 장치의 레지스터를 통하여 명령어가 저장된 시작 주소와 크기를 알려 준다. 제어 장치의 상태 천이기는 시작 번지에 있는 명령어를 인출하여 처리기 모듈로 전달하고, 다음 명령어를 가져오기 위해 주소를 증가시킨 후 처리기 모듈의 상태 천이기의 진행 상태를 플래그(flag) 비트를 이용하여 확인한다. 상태 플래그 비트가 활성화되면 곧바로 증가된 주소의 다음 명령어를 인출하여 처리기 모듈로 전달한다.

명령어 큐는 32 bits로 구성하여 메모리 접근에 대한 명령어와 일반 명령어들을 동시에 인출함으로써 지연 시간을 줄이도록 하였고, 버스 중재기를 두어 프로세서와 제어 장치가 동시에 접근을 시도할 때 충돌이 발생하지 않도록 하였다.

4.5 처리기 모듈

처리기 모듈은 병렬처리 시스템에서 가장 중요한 부분이다. 하나의 처리기 모듈은 구현될 FPGA의 크기를 고려하여 4개의 처리기를 가지도록 하였으며, 기본적으로 SIMD 구조를 가지고 있다. 처리기 모듈은 하나의 상태 천이기와 4개의 처리기로 되어 있으며, 각 처리기는 앞에서 정의한 모든 명령어 집합을 지원한다.

상태 천이기는 제어장치로부터 전달된 명령어를 해석하고 수행하는 주체 역할을 하는 부분으로, 메모리 접근을 위한 상태 천이기와 산술·논리 연산 및 조건 검색과 같은 일반 명령어 수행을 위한 상태 천이기로 구분되어 있다. 일반 명령과 메모리 전송 명령이 동시에 수행되기 때문에 두 개의 상태 천이기는 플래그 비트를 이용하여 제어 장치와 서로 동기화를 유지하도록 되어 있다. 각각의 처리기는 하나의 산술논리 연산기와 4개의 범용 레지스터, 2개의 특수목적 레지스터를 가지고 있는데, 2개의 특수목적 레지스터는 i 번째 메모리 접근 명령 수행의 결과가 ($i+1$)번째 일반 명령어 수행에 적용되기 위한 레지스터(8bits)와 모든 산술 연산을 signed-2's complement 형태로 수행하지만 실제적으로 처리될 영상 데이터를 256-레벨 영상으로 고려하였기 때문에 unsigned magnitude 형태가

접목된 연산을 수행할 수 있도록 만들어진 레지스터(16bits)가 있다.

다중접근 기억장치는 그림 3에 나타난 바와 같이 처리기로부터 전송된 기준 좌표 X, Y와 간격 신호를 이용하여 m개의 기억 모듈로부터 pq개의 기억모듈을 선택하는 모듈선택 회로, 새로운 주소를 계산하는 주소계산 회로, 그리고 계산된 주소와 전송된 데이터를 원하는 기억 모듈에 맵핑시키기 위한 주소라우팅 회로 및 데이터 라우팅 회로로 구성된다. 각각의 회로들은 그 복잡도에 따라 지연 시간이 다르기 때문에 레지스터를 추가하여 시간차를 보상하도록 하였다.

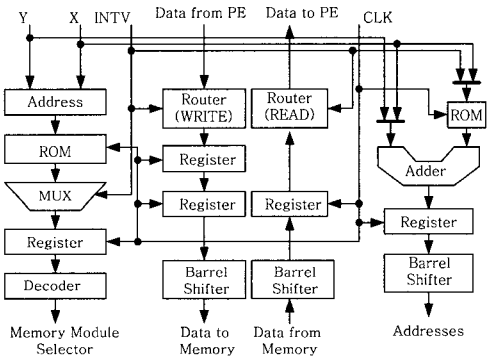


그림 3. 다중접근 기억장치의 블록도

Fig 3. Block diagram of Multi-access memory system

4.6 메모리 모듈

메모리 모듈은 128KB x 8 bits 형태의 5(= $pq+1$)개 SRAM으로 구성하였으며, 메모리 접근 시간은 병렬처리 시스템의 성능을 좌우할 수 있는 부분이기 때문에 1 클럭에 접근할 수 있는 메모리를 선정하여 설계하였다.

V. 설계 검증 및 모의실험

병렬처리 시스템에서 영상 처리를 수행하기 위해서는 수행할 알고리즘을 제안된 시스템의 구조에 맞게 재설계하는 과정이 필요하다. 즉, 알고리즘을 여러 개의 처리기가 수행하기 때문에 입력 영상을 분할하여 각각의 처리기에 할당하고 처리기가 지원하는 기계어(machine code)로 프로그램을 작성해야 한다.

제안한 병렬처리 시스템의 성능을 검증 및 분석하기 위한 응용 프로그램으로 영상 신호로부터 물체의 구조적인 정보를 추출하는데 매우 효과적인 도구이며 수리 형태론에 기반을 둔 형태학적 필터(morphological filters)를 선정하였다. 수리 형태론의 기본적인 연산자에는 erosion과 dilation이 있으며 이들은 형태소(structure element)라 불리는 연산의 탐침 역할을 하는 작은 영상을 이용하여 연산을 수행한다. 특히, 수리 형태론 연산자들을 이용한 필터는 기존의 다른 필터와는 달리 기하학적 형상 구조에 바탕을 두고 영상을 처리함으로써 영상에서 기본적인 정보는 보존하고 동시에 불필요한 요소는 제거하는 특성이 있어 영상 분할을 위한 알고리즘의 전처리 부분에서 좋은 결과를 제공해 주지만 수행시간이 오래 걸리는 단점을 지니고 있다. 그래서, 제안한 병렬처리 시스템의 성능 분석을 위한 응용 알고리즘으로 선정하였다.

우선, 처리해야 할 입력 영상이 $m \times n$ bytes로 구성되어 있을 때, 두 조각으로 나누어 각 처리기 모듈의 다중접근 기억장치에 저장되는데, 크기를 $m \times (\lfloor n/2 \rfloor + 1)$로 하여 두 조각이 서로 겹치도록 한다. 이것은 분할된 영상의 경계 부분에 대한 연산을 수행할 때 처리기 모듈간의 통신을 없애기 위한 것이며, 겹치는 부분의 크기는 형태소의 크기에 따라 달라질 수 있다.

처리기 모듈 내의 각 처리기는 주어지는 간격에 따라 처리할 데이터가 정해지는데, 그림 4-(a)는 13 x 11로 구성된 입력 영상을 분할하여 처리기 모듈에 할당하는 방법을 나타내며, 간격이 '1'인 경우 각 처리기가 수행할 데이터들이 저장될 메모리 모듈에 대한 예는 그림 4-(b)와 같다.

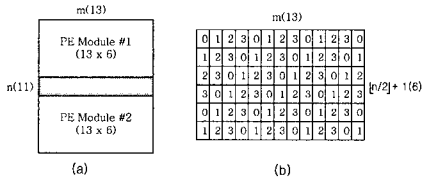


그림 4. 메모리에 저장된 데이터를 처리기 모듈에 할당하는 예

Fig 4. An example of raw image assignment to PEs

다음은 수행해야 할 알고리즘을 병렬화하고 기계

어로 변환하는 과정이 필요한데, 반드시 메모리 접근 명령어와 산술·논리 연산에 대한 일반명령어가 쌍을 이루도록 해야 한다. 다음은 256-레벨 영상에 대한 erosion과 dilation에 대한 예제 프로그램을 나타낸 것으로서 하나의 데이터에 대한 연산이므로 이와 같은 프로그램을 기준좌표 X, Y만 변경하면서 처리기에 할당된 데이터 수만큼 반복한다.

(a) Erosion

메모리 접근 명령어	일반 명령어
READ (1, 0, 0)	VALTRAN AC 256
READ (1, 1, 0)	COND1 RD_REG
READ (1, 2, 0)	COND1 RD_REG
...	...
NOP	COND1 RD_REG
WRITE (1, 1, 1)	NOP

(b) Dilation

메모리 접근 명령어	일반 명령어
READ (1, 0, 0)	VALTRAN AC 0
READ (1, 1, 0)	COND2 RD_REG
READ (1, 2, 0)	COND2 RD_REG
...	...
NOP	COND2 RD_REG
WRITE (1, 1, 1)	NOP

처리기 블록은 Verilog-HDL을 사용하여 설계하였으며 기능 검증을 위하여 CADENCE사의 Verilog-XL을 이용하여 1차 모의실험을 수행하였다. 기능 검증을 완료한 후 ALTERA 사의 Maxplus II Tool에서 FLEX 칩을 타겟으로 컴파일하여 Verilog-HDL로 된 delay 데이터를 얻고 Back annotate하여 모의실험을 완료하였다. 그림 5에서 Verilog-XL로 모의실험한 결과를 파형으로 보여주고 있으며, 파형에서 나타난 각 신호들은 다음과 같다.

- c_state_cu : 명령어 인출을 담당하는 제어장치의 상태 천이기를 나타내는 신호이다.

- en_mem, en_opr : 제어 장치에서 생성되어 처리기 모듈로 전해지며, 메모리 접근과 산술·논리 연산을 위한 상태 천이기의 시작 플래그로 Active high 신호이다

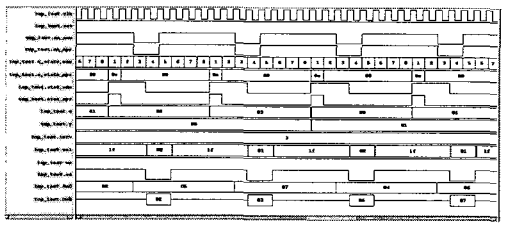


그림 5. 형태학적 필터를 수행 중인 제안한 병렬처리 시스템의 모의실험 파형

Fig 5. Waveform during simulation of the proposed parallel processing system applied to morphological filter

- c_state_mem : 처리기 모듈의 메모리 접근을 위한 상태 천이기를 나타내는 신호이다.

- c_state_opr : 처리기 모듈의 산술·논리 연산을 위한 상태 천이기를 나타내는 신호이다.

- stat_mem, stat_opr : 메모리 접근과 산술·논리 연산을 위한 상태 천이기의 상태 플래그로 처리기 모듈에서 생성되어 제어 장치로 전달되며 Active high이다.

- X, Y, intv : 처리기에서 다중접근 기억장치로 전송되는 기준좌표 X, Y와 간격 신호이다.

- sel, we, oe : 다중접근 기억장치 내의 메모리 모듈의 선택, 쓰기, 그리고 읽기를 제어하는 신호로 Active low이다.

- Am[x], Dm[x] : 다중접근 기억장치 내의 메모리 모듈을 접근하기 위한 주소 및 데이터 신호를 나타내는 신호이다.

파형의 결과를 살펴보면 명령어 인출과 메모리 접근 및 산술·논리 연산의 수행이 중첩되어 진행되고 있으며, 수행 시간이 가장 긴 메모리 모듈의 접근 속도에 따라 전체 수행 시간이 결정됨을 알 수 있다. 따라서, 앞의 예제 프로그램으로 보인 256-레벨 영상에 대한 erosion과 dilation을 수행하는데 걸리는 시간은 다음과 같이 계산될 수 있다.

형태소의 크기가 3x3일 경우, 모두 9번의 메모리 모듈 읽기 접근과 1번의 추가적인 비교 연산, 그리고

1번의 메모리 모듈 쓰기 접근에 의해 수행 시간이 결정된다. 메모리 모듈의 읽기(T_{read})는 8클럭, 쓰기 (T_{write})는 6클럭을 소모하고 비교 연산(T_{cond})은 2클럭이 필요하므로 33MHz로 동작할 때 하나의 데이터에 대해서 소요되는 시간은,

$$\begin{aligned} T_{total} &= 9\times T_{read}+1\times T_{cond}+1\times T_{write} \\ &= 9\times 8\times 30(ns)+2\times 30(ns)+6\times 30(ns) \\ &= 2400(ns). \end{aligned}$$

즉, 하나의 처리기가 하나의 데이터에 대한 erosion과 dilation 연산을 수행하는데 걸리는 시간은 2.4(μs)이다. 같은 방법으로 계산하면 형태소의 크기가 5x5인 경우는 수행시간이 6.24(μs)가 소요됨을 알 수 있다.

병렬처리 시스템의 비교 대상을 동작속도 및 메모리 접근 속도가 제안한 시스템과 유사하고 시간 측정을 위한 타이머를 가지고 있는 인텔사의 i960RM SDK 보드와 이전에 설계된 병렬처리 시스템[16]-[18]으로 선정하였다. i960RM 프로세서의 내부 클럭은 100MHz, 버스 클럭은 33MHz로 동작하며, 메모리는 66MHz SDRAM을 사용한다. 16KB의 명령어 캐쉬와 4KB의 데이터 캐쉬를 가지고 있으며, 프로세서 내부에 10ns 단위로 측정할 수 있는 타이머를 가지고 있다. 프로그램은 'C'언어로 작성하였으며, 명령어 캐쉬와 데이터 캐쉬를 ON 상태로 설정하고 측정하였다.

이전의 병렬처리 시스템에서는 3x3의 형태소를 사용하여 256-레벨 영상에 대한 erosion과 dilation을 수행할 때에는 9번의 메모리 읽기 접근(T_{read})과 비교 연산(T_{cond}), 그리고 1번의 메모리 쓰기 접근 (T_{write})이 수행된다. 이 때 명령어 인출(T_{instr})은 5클럭에 가능함으로 하나의 데이터에 대한 연산 소요 시간은

$$\begin{aligned} T_{total} &= 19\times T_{instr}+9\times T_{read} \\ &\quad +9\times T_{cond}+1\times T_{write}+\alpha \\ &= 19\times 5\times 30(ns)+9\times 8\times 30(ns) \\ &\quad +9\times 2\times 30(ns)+6\times 30(ns) \\ &= 5730(ns) \end{aligned}$$

즉, 처리기 하나가 하나의 데이터에 대한 erosion이나 dilation을 수행할 때 걸리는 시간은 5.73(μs)이다. 위에서 α는 구현상에서 발생할 수 있는 프로세서와 처리기 모듈간의 동기화를 위해 소요되는 시간으로 여기서는 제외하였다. 같은 방법으로 형태소의 크기를 5x5로 하였을 때에는 수행시간이 15.48(μs)가 소요되는 것으로 측정되었다.

표 1. i960RM SDK에서 측정한 수행 시간
Table 1. Estimated processing times on i960RM SDK

입력 영상	SE Size	Erosion (sec)	Dilation (sec)	Difference (sec)	Edge Detection (sec)
QCIF(176x144)	3x3	0.280880	0.263760	0.075440	0.620080
"	5x5	0.928400	0.912240	0.075440	1.916080
CIF(352x288)	3x3	1.136800	1.067760	0.301680	2.506240
"	5x5	3.807440	3.740800	0.301680	7.849920

표 1은 352x288 크기의 CIF 영상과 176x144 크기의 QCIF 영상에 대해서 3x3 크기의 형태소와 5x5 크기의 형태소에 의해 erosion과 dilation 및 경계선 검출 알고리즘을 i960RM SDK에서 수행한 시간을 측정한 결과이며, 표 2는 이전의 병렬처리 시스템에서 처리기를 8개로 가정하여 같은 연산을 수행하였을 때 소요되는 시간을 계산한 결과이다.

제안한 병렬처리 시스템은 두 개의 처리기 모듈, 즉, 8개의 처리기를 가정하였으며, 참조 시스템들과 같은 33MHz로 동작했을 때의 결과이다. 병렬처리 시스템에서는 처리할 입력 영상을 다중접근 기억장치로 가져오고, 처리된 결과를 호스트 컴퓨터로 전송하는 과정이 필요하므로 그에 따른 측정된 소요 시간은 표 3과 같다.

표 3. 호스트 컴퓨터와 병렬처리 시스템간의 데이터

표 2. 이전에 설계된 병렬처리 시스템에서 측정한 수행시간

Table 2. Estimated processing times on the previously designed parallel processing system

입력 영상	SE Size	Erosion (sec)	Dilation (sec)	Difference (sec)	Edge Detection (sec)
QCIF(176x144)	3x3	0.018153	0.018153	0.004847	0.084776
"	5x5	0.049040	0.049040	0.004847	0.102927
CIF(352x288)	3x3	0.072610	0.072610	0.019388	0.164608
"	5x5	0.196162	0.196162	0.019388	0.411712

전송 시간

Table 3. Estimated transfer times between host computer and the proposed system

입력 영상	전송 시간(WRITE)	전송 시간(READ)
QCIF(176x144)	1.140480(ms)	1.520640(ms)
CIF(352x288)	4.561920(ms)	6.082560(ms)

그리고, 제안한 시스템에서 수행한 결과는 표 4와 같다.

표 4. 제안한 병렬처리기에서 측정한 수행시간

Table 4. Estimated processing times on the proposed system

입력 영상	SE Size	Erosion (sec)	Dilation (sec)	Difference (sec)	Edge Detection (sec)
QCIF(176x144)	3x3	0.0076032	0.0076032	0.002249	0.017455
"	5x5	0.0197683	0.0197683	0.002249	0.041786
CIF(352x288)	3x3	0.0304128	0.0304128	0.008997	0.069823
"	5x5	0.0790732	0.0790732	0.008997	0.167143

256-레벨 영상에 대한 erosion과 dilation 연산을 토대로 경계 검출 알고리즘을 수행한 결과를 그림 6에서 그래프로 나타내었다. 그림 6에서 나타난 결과는 최초에 호스트 컴퓨터로부터 영상 데이터를 가져오고 수행한 결과를 호스트로 전송하는 시간까지 포함한 결과이다. 이 그래프에서 알 수 있듯이 기존의 일반 컴퓨터에서 순차적으로 처리하는 것에 비해서 다중접근 기억장치를 이용한 병렬처리 시스템에서 수행하였을 때에는 약 30~45배 정도의 시간이 절약됨을 알 수 있다. 이런 큰 차이는 일반 컴퓨터 시스템에서는 영상 데이터를 순차적으로 처리할 뿐만 아니라 연산을 위한 프로그램 수행과 그에 따른 메모리 접근이 추가적으로 일어나기 때문인 것으로 보인다. 그리고, 이전에 설계된 병렬처리 시스템에 비해서는 약 3배 정도의 시간이 절약되는 것으로 나타났다.

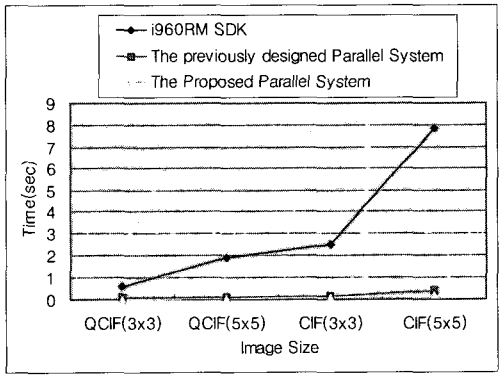


그림 6. 실험 결과 : 256-레벨 영상에 대한 경계검출 알고리즘의 수행시간

Fig 6. Experimental results : Times yielded by processing edge detection on 256-level images

VI. 결 론

다중접근 기억장치를 이용한 SIMD 형태의 새로운 병렬처리 시스템의 구조를 제안하고 설계하였으며, 영상 처리에서 많이 사용되는 수리 형태론적 연산을 적용하였을 때 일반 컴퓨터 시스템과 기존에 설계된 병렬처리 시스템에 비해 처리 속도가 월등하게 향상됨을 입증하였다.

본 논문에서는 병렬처리 시스템의 응용을 256-레벨 영상에 대한 수리 형태론적 연산에 국한하였으나, 영상처리에 적용되는 대부분의 알고리즘이 이와 유사하게 이루어짐으로 병렬처리 시스템에 쉽게 적용이 가능하다.

본 논문에서 제안한 병렬처리 시스템을 ALTERA 사의 FPGA에 fitting 하였는데, 디바이스 특성으로 인하여 동작 주파수를 33MHz로 제한하였다. 추후 ASIC 형태로 구현하여 동작 주파수와 병렬도를 높인다면 비디오 영상에 대해 자동 영상 분할과 같은 일부 알고리즘의 실시간 처리가 가능할 것으로 보인다.

제안한 병렬처리 시스템은 PCI 버스를 지원하는 일반 컴퓨터에 적용하는 것이 목표이다. 따라서, 일반 컴퓨터에서 운용되는 운영체제에 맞는 드라이버 개발과 응용 프로그램 및 그래픽 사용자 인터페이스 개발이 이루어져야 한다. 또한, 병렬처리 시스템을 다양한 응용에 적용하기 위한 병렬 알고리즘의 개발이 필요하다. 그리고, 성능 개선을 위해서 pipeline 단계를 좀 더 세분화하여 메모리 접근에 소요되는 시간을 줄이고, 호스트 컴퓨터와의 데이터 전송 방법을 개선한다면 성능면에서 많이 향상될 것으로 판단된다.

VII. 참고문헌

[1] J. L. Potter, "Image processing on the massively parallel processor," *IEEE Computer*, vol. 16, no. 1, pp. 62-67, Jan. 1983

[2] A. Rosenfeld, *Multiresolution image processing and analysis*, Springer-Verlag, 1984

[3] H. S. Wallace and D. Howard, "HBA vision architecture," *IEEE Trans. PAMI*, vol. 11, no. 3, pp. 227-232, Mar. 1989

[4] M. Duff, Parallel processors for digital image processing, *Advances in Digital Image Processing. P. Stucki, Ed.* NewYork: Plenum, 1979, pp.265-279.

[5] K. E. Batcher, Design a massively parallel processor, *IEEE Trans. Comput.*, Vol. C-29, pp. 836-840, Sep. 1980

[6] M.J. Kimmel, et al., MITE: Morphic image transform engine, an architecture for reconfigurable pipelines of neighborhood processors, *in Proc. IEEE Comput. Soc. Workshop Comp. Architecture for Pattern Analysis and Image Database Management*, Miami Beach, FL, Nov. 18-20, 1985, pp.493-500.

[7] E. W. Kent, et al., Hierarchical cell logic and the PIPE processor: structural and functional correspond-ence, *in Proc. IEEE Comput. Soc. Workshop Comp. Architecture for Pattern Analysis and Image Database Management,* Miami Beach, FL, Nov. 18-20, 1985, pp.311-319.

[8] P. Budnik and D. J. Kuck, "The Organization and Use of Parallel Memories," *IEEE Trans. Computers*, vol. C-20, no. 12, pp. 1566-1569, Dec. 1971

[9] D. H. Lawrie, "Access and Alignment of Data in Array Processor," *IEEE Trans. Computers*, vol. C-24, no. 12, pp. 1145-1155, Dec. 1975

[10] D. Lee, "Scrambled storage for parallel memory systems," *In Proc. Int. Symp. on Comp. Architecture*, pp. 232-239, 1988

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

[12] D. H. Lawrie and C. R. Vora, "The Prime Memory System for Image Processing," *IEEE Trans. Computers*, vol. C-31, no. 5, pp. 435-442, May 1992

[13] C. S. Raghavendra and R. Boppana, "On methods for fast end efficient parallel access," *In Proc. Int. Parallel Processing*, pp. 76-83, 1990

[14] J. W. Park, "An Efficient Memory System for Image Processing", *IEEE Trans. Computers*, vol. C-35, no. 7, pp.669-674, Jul. 1986

[15] J. W. Park, D. T. Harper III, "An Efficient Memory System for the SIMD Construction of a Gaussian Pyramid." IEEE Trans. on Parallel and

Distributed System, Vol. 7, No. 7, pp. 855-860, July 1996

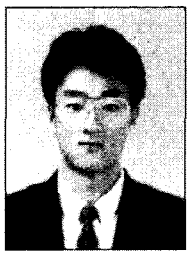
[16] K. A. Moon, H. Lee, J. W. Park, "A parallel processing system for a high-speed printed document recognition," *MVA'96 IAPR Workshop on Machine Vision Applications*, pp. 518-521, Nov. 1996

[17] 윤희준, 이형, 한기선, 박종원,"문화재 검색을 위한 병렬처리기 구조," *한국멀티미디어학회 논문지*, 제 1권 2호,154-161, 1998년 12월

[18] Hyung Lee, Kyung-Ae Moon, and Jong-Won Park, "Design of Parallel Processing System for Facial Image Retrieval," *ACPC 99 Parallel Computation, Springer-Verlag*, pp.592 ~ 593, Feb. 1999

저 자 소 개

李 炯 (學生會員)



1995년 2월 : 충남대학교 컴퓨터과학과 졸업
1997년 2월 : 충남대학교 컴퓨터공학과 졸업, 공학석사
2000년 2월 : 충남대학교 컴퓨터공학과 박사과정 수료. 관심분야 : 영상처리, 병렬처리, 반도체 설계

金 重 培 (會員申請中)
1989년 2월 : 한남대학교 전자공학과 졸업
1989년 2월 : 전자통신연구원
2000년 8월 : 충남대학교 정보통신공학과 졸업, 공학석사
2000년 8월 ~ 현재 : 욱성전자(주) 이사
관심분야 : 컴퓨터 구조, 반도체 설계

崔 成 赫 (會員申請中)

1991년 2월 : 한국항공대학교 정보통신공학과 졸업
1991년 2월 ~ 현재: 전자통신연구원 연구원
1999년 2월 : 충남대학교 전자공학과 졸업, 공학 석사
1999년 3월 : 경북대학교 전자공학과 박사과정 입학.관심분야 : 병렬처리, 데이터 통신

朴 宗 元 (正會員)

1979년 2월 : 충남대학교 전자공학과 졸업. 1981년 2월 : 한국과학기술원 전산학과 졸업, 공학석사
1991년 8월 : 한국과학기술원 전산학과 졸업, 공학박사. 1995년 ~ 현재 : 충남대학교 공과대학 정보통신공학교 정교수. 관심분야 : 영상처리, 병렬처리, 의공학