論文 92-29B-4-1

병렬컴퓨터 "KAPAC"의 설계 및 구현

(Design and Implementation of a Parallel Computer "KAPAC")

成 桐 洙*, 姜 輝 三**, 崔 勝 旭***, 朴 圭 皓*

(Dong Su Seong, Whi Sam Kang, Seung Uk Choi, and Kyu Ho Park)

要 約

트랜스퓨터를 근간으로 하는 병렬컴퓨터 "KAPAC(KAist PArallel Computer)"을 설계하고 구현하였다. KAPAC의 목적은 복잡하거나 많은 계산이 요구되는 일을 병렬로 처리하여 속도 향상을 시킴으로써 실시간 처리및 고성능 처리를 하는 많은 응용분야에 대한 계산능력을 제공하기 위함이다. KAPAC은 UNIX 컴퓨터를 Host로 하고 VME bus에 연결할 수 있는 후위 컴퓨터로 구현하였다. 구현된 병렬 컴퓨터는 32개의 처리소자를 가지고 있는 메세지 패싱 타입의 컴퓨터이며 크로스바 스위치를 사용하여 프로그램에 의해 쉽게 연결망 형태를 구성 할 수 있도록 하였다. 구현된 병렬 컴퓨터 "KAPAC"의 재구성 특성을 보기 위하여 구성할 수 있는 다양한 연결망들을 소개했으며 몇개의 응용 프로그램들이 각기 다른 상호 연결 위상에서 수행되었다.

Abstract

A parallel computer "KAPAC(KAIST Parallel Computer)" based on Transputer is designed and implemented. its purpose is to support the real time processing and high perfomance computing through parallelizing the complex and heavy computation load. KAPAC has UNIX machine as host-computer and is implemented on VME bus as back-end machine. The parallel computer "KAPAC" is the message-passing loosely-coupled multiprocessor computer having thirty two processing elements, and the network topology between processing elements can be easily configured with the crossbar switchs using the control program. Various topologies are introduced and appoication programs are executed on the parallel computer "KAPAC" with eifferent interconnection topologies to show the reconfigurability.

I. 서 론

최초의 아날로그 컴퓨터가 1883년에 개발되고 최초의 디지탈 컴퓨터 ENIAC이 1946년에 개발된 이후 계전기, 진공관, 트랜지스터, 집적회로로 이루어는 스위칭 소자의 발전과 소프트웨어의 발전으로 디지탈 컴퓨터는 속도, 구조, 제공하는 서비스 측면에서 많은 발전을 해왔다. 그런데 초창기에는 단일 프로세서를 사용한 Von Neumann 방식의 순차적(sequential) 컴퓨터가 대부분이었으나 1970년대 초반에 Illiac IV와 C.mmp 같은 병렬처리 컴퓨터들이 소개되기 시작하였다. 현재는, 집적회로 기술의 획기적인 발달로 단일 프로세서의 계산 능력이 획기적으로 강력하여 졌으며, 이를사용하여 과학및 공학 분야등에서의 점증하는 계산 용량을 만족시키기 위한 방법으로 병렬처리에 대한 연

*正會員, 韓國科學技術院 電氣 및 電子工學科
(Dept. of Electrical Eng., KAIST)
**正會員, 三星電子 家電研究所
(Samsung Research Center)
***正會員, 韓國通信 研究開發團
(Korea Telecom Research Center)
接受日字: 1991年 11月 2日

구가 활발히 진행되고 있다.

지금까지 등장한 병렬처리 시스템은 pipe-line 컴퓨터, array processor, multiprocessor 시스템, 그리고 새로운 개념의 병렬처리 컴퓨터인 data-flow 컴퓨터, systollic array 등으로 구분된다.[1,2,3] 이중에서 multiprocessor 시스템은 명령과 데이타의 병렬성 정도에 따른 Flynn[4]의 분류법에 의하여 MIMD (multiple instruction streams multiple data streams) 컴퓨터 구조에 속하는데, 통신 방식에 따라 프로세서간 메모리를 공유하여 정보 교환을 하는 shared-memory 시스템과 프로세서간 직접적인 통신 링크를 통해서 정보 교환을 하는 message-passing시스템이 있다.[5,6] 전자를 tightly-coupled multiprocessor시스템이라 하고 후자를 loosely couple multiprocessor시스템 이라고 한다.[7,8,9,10] Shared-memory multiprocessor 시스템은 통신속도가 메모리의 대역폭(bandwidth)에 의해서 제한된다. 따라서 프로세서의 수가 증가하면 memory contention에 의해서 전체 시스템의 효율이 감소하게 된다. 반면 message-passing multiprocessor 시스템은 각 프로세서가 자기만의 메모리를 가지고 있어서 memory contention같은 현상은 없지만 정보 교환을 위한 비용이 상대적으로 많아진다. 그러므로 이 방식은 프로세서간 통신이 많지 않거나 프로세서의 수가 매우 많을 때 효율적이다.[11,12]

본 논문에서 설명하고자하는 병렬 컴퓨터 "KAPAC"은 메시지 패싱 타입의 컴퓨터에 속하며 UNIX Host 컴퓨터의 후위 컴퓨터(back-end computer)로 구현하였다. KAPAC은 32개의 처리 소자를 갖고 있으며, 이 처리 소자들의 연결구조를 소프트웨어에 의하여 제어할 수 있도록 하기 위하여 reconfigurability를 갖고 있는 크로스바 스위치들이 사용되었다. 또한 모든 처리소자와 네트워크를 관장하기 위한 마스터 부분과 처리소자로 이루어진 슬레이브 부분으로 되어 있으며, Host 컴퓨터와 마스터와의 통신은 VME bus를 통하여 이루어 진다. 슬레이브는 8개의 처리소자를 가진 네개의 슈퍼노드(supernode)[13] 구조로 구성하여 modularity 특성을 갖도록 했다. 이들 마스터 부분과 슈퍼노드 부분은 각각 한장의 보드로 만들었으며, 또한 이들 사이의 필요한 연결을 하기 위한 연결 보드를 backplane 형태로 만들어 손쉽게 사용할 수 있도록 하였다. 그리고 중앙 집권적인 관리가 되도록 구성함으로써 처리 소자들을 여러 영역으로 분할하여 독립된 일의 수행이 가능하도록 하였다. 각 처리소자를 구성하기 위해 INMOS사의 트랜스퓨터를 이용하였는데 이는 Hoare의 CSP (communicat-sequenti processes)[13,14]에 기초를 두는 OCCAM 프로그래밍 언어의 모델을 기반으로 하여 on-chip하드웨어로 병렬처리를 지원하는 마이크로 프로세서이다. 트랜스퓨터는 하나의 프로세서와 그 프로세서에 의해 실행될 프로그램을 저장하기 위한 메모리(on-chip memory) 그리고 다른 트랜스퓨터와의 일대일 연결을 위한 네개의 통신용 링크 및 수치계산을 위한 FPU(floating point unit)로 구성되어 있다.[15]

본 논문은 크게 KAPAC의 하드웨어와 소프트웨어에 대한 부분으로 구성되는데, Ⅱ장에서 본 병렬 처리 컴퓨터 KAPAC의 하드웨어 구조 및 특성을 설명하며, Ⅲ장에서 device driver, file server, 그리고 응용 프로그램 등과 같이 KAPAC에서 필요한 소프트웨어를 설명한 다음, Ⅳ장에서 성능 평가의 결과를 비교 및 검토하고 마지막 Ⅴ장에서 결론을 맺는다.

Ⅱ. 하드웨어

1. 개 요

KAPAC의 전체 구성은 그림 1과 같으며 크게 HM (host module)과 CM (computing module)으로 나누어진다. HM은 사용자와의 모든 interface를 제공하는 역할 및 자료(data나 실행 file) 저장 역할, 그리고 CM을 제어하는 controller 역할등을 담당하는 부분으로 MVME 147 cpu board등으로 구성되어 있고, CM은 실제 병렬 처리를 할 수 있도록 하는 부분인데, VIU (VME interface unit)와 PPU (parallel processing unit)로 이루어져 있다. VIU는 HM과 PPU사이의 자료 교환이나 HM으로 부터 정보를 받고서 PPU을 제어할 수 있도록 신호를 생성하는 역할을 한다. 이 부분은 크게 INMOS C012 chip으로 이루어진 link adaptor, DMA Controller, VME bus requester, VME interrupter 등으로 구성되어 있다. 마지막으로 KAPAC에서 가장 핵심적인 부분인 PPU는 하나의 RPE (root preprocessing element) 및 하나의 CPE (control processing element) 그리고 최대 32개의 NPE (node processing element)로 이루어진 처리 부분및 이들의 상호연결을 담당하는 연결부분 그리고 각 node processing element를 제어하고 오류 발생 검사를 하기 위한 부분으로 이루어져 있다. 이 CM의 전체 구성은 그림 2와 같으며 상세한 설명은 2.4절에서 설명 할 것이다.

KAPAC의 구조적 특징은 32개의 처리소자(processing element)간에 임의의 연결을 이룰 수 있고, 소프트웨어에 의해 프로그램내에서도 자유자재로 바꿀 수 있는 동적 변환 방식을 사용하므로써 각 사용자들이 임의의 병렬 구조를 선택하여 사용할 수 있는 효용성을 가질 수 있다. 또한 여러 영역으로 나누어

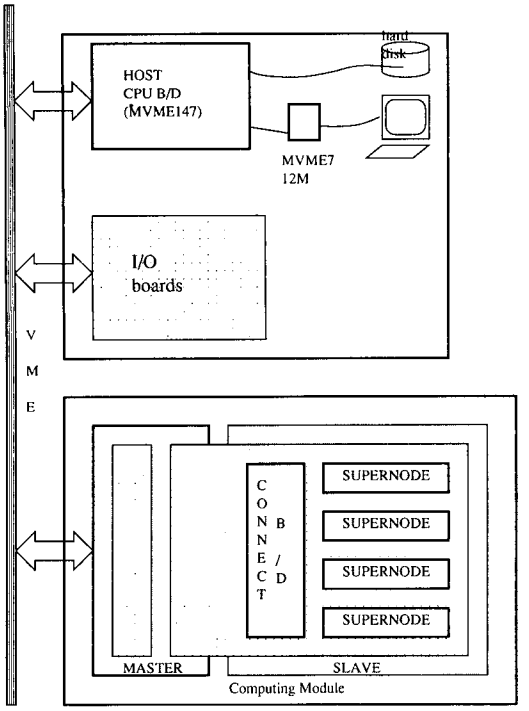

그림 1. 병렬컴퓨터 "KAPAC"의 전체 구성도
Fig. 1. Overall configuration of parallel computer "KAPAC."

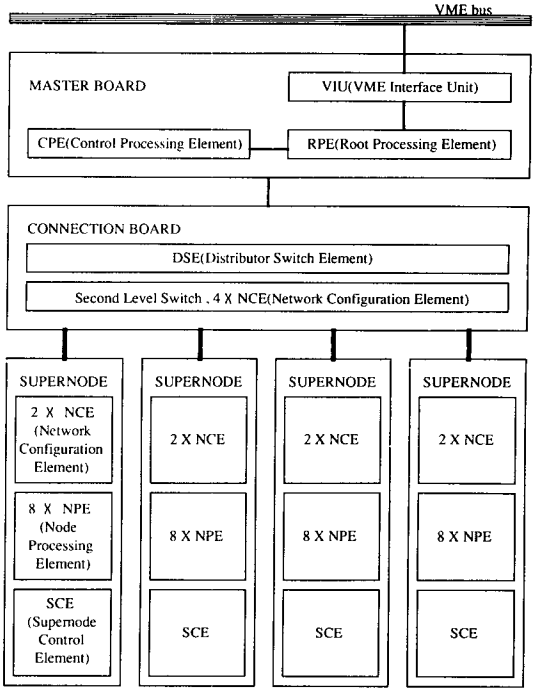

그림 2. 컴퓨팅 모듈의 하드웨어 구성도
Fig. 2. Hardware configuration of computing module.

사용할 수 있도록 partitionable하고 필요에 따라 계산 능력을 조정할 수 있도록 modular하게 설계 되었으며 상호 연결망을 구성하는 스위치 소자와 각 처리 소자를 감시하는 제어 소자를 중앙 집권적으로 조정할 수 있도록 구현되어 있다. 우선 본 KAPAC를 위하여 사용한 주요 chip들을 간략히 설명하고 위에 언급한 각 부분에 대하여 논한다.

2. KAPAC에 쓰인 주요 chip들의 소개

KAPAC에 쓰인 주요 chip들은 크게 INMOS 사에서 제공되는 IMS T800 및 IMS C004 crossbar switch, IMS C012 link adaptor가 있고, Motorola사에서 제공되는 MC68450 DMA controller가 있으며 그 중 T800과 C004에 대해 설명한다.

1) IMS T800

KAPAC에서 프로세싱 소자로 사용하고 있는 T800 트랜스퓨터는 INMOS사에서 개발한 32bit 마이크로프로세서로 4개의 직렬 링크가 있고 고속의 프로세싱을 위하여 내부에 4Kbyte의 RAM을 가지고 있는 고성능 프로세서로서, 64bit floating point unit 및 H/W scheduler를 가지고 있기 때문에 고속도 처리 및 병렬프로그래밍에 적합하다.

2) IMS C004

처리소자들은 다른 처리소자들과의 통신을 위하여 4개의 통신 링크가 있다. 그러나 4개의 통신 링크를 사용하여 다른 처리소자와 직접 연결시켜 network을 구성할 경우 인접하지 않는 처리소자와의 통신을 위해서는 중간단계에 있는 처리소자의 중계가 필요하다. 많은 처리소자를 가진 시스템의 경우 임이의 처리소자간 통신을 위하여 많은 시간이 소요되며 또 연결구조가 고정되어 있다는 단점이 있다. 이를 위해 INMOS사에서는 각 처리소자간의 연결을 소프트웨어에 의해 직접 연결 할 수 있는 32×32 C004 크로스바 스위치를 제공하고 있다.

3. Host Module (HM)

KAPAC에서 CM을 제외한 나머지 부분으로 전체 시스템의 호스트 역할을 하는 CPU 보드, Host용 운영체제 및 CM을 위한 자료등을 저장하기 위한 hard disk, 사용자 interface를 제공하는 terminal로 기본구성을 이루고 있다. 하지만 특별한 목적을 위해서 다른 보드를 추가할 수 있다. 예를들어, 본 KAPAC의 경우에는 카메라로 부터 영상을 얻고 모니터에 출력

하기 위하여 FG-100V 보드를 사용하였다. HM에서 CPU 보드는 MC 68030 CPU를 사용하였으며, 운영체제로는 UNIX system V를 이용하였다.

4. Computing Module (CM)

1) VIU (VME Interface Unit)

HM과 PPU사이의 자료 교환이나 HM이 PPU를 제어할 수 있도록 하기 위해서 필요하며 이 부분의 제어나 상태 레지스터들이 HM 컴퓨터의 메모리 영역중 VME bus[17] 상의 한 영역에 매핑(mapping) 되도록 하기 위하여 I/O mapped I/O 방식을 이용하였다. 그리고 PPU와는 INMOS C012 link adapter와 몇개의 신호선에 의해서 상호 연결되어있으며 제공하는 HM컴퓨터와 PPU사이의 통신 방식은 polling 혹은 DMA[16]에 의한다. 그리고 polling시 handshake를 위해 필요한 상태검사를 하는 방법으로 컴퓨터가 polling하는 방법과 CM에서 interrupt 신호에 의한 방법이 있다. 따라서 HM과 CM과의 통신방법은 세가지로 세분할 수 있다. 그 전체 block diagram은 그림3에 나타나 있다.

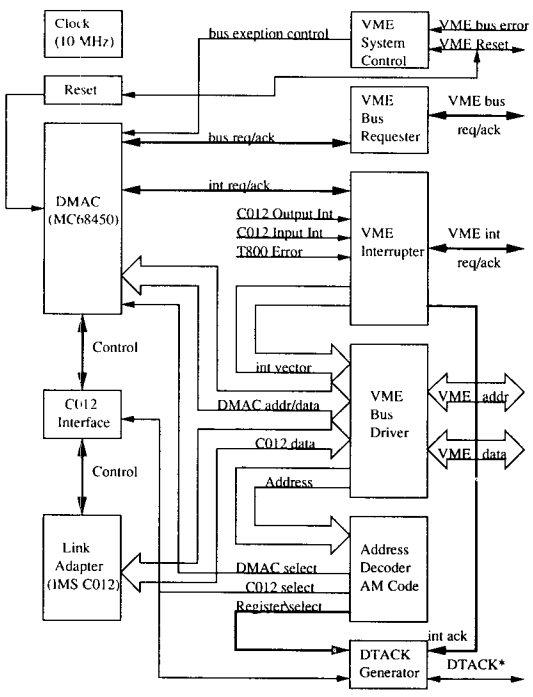

그림 3. VME interface unit의 block diagram
Fig. 3. Block diagram of VME interface unit.

2) PPU (Parallel Processing Unit)

PPU는 CM (computing module)에서 VIU (VME Interface unit)를 제외한 나머지 부분으로 실제 병렬처리를 지원하는 부분이며 32개의 PE (processing element)들을 포함하고 있다. 이 PE들은 supernode 구조로 8개씩 clustering되어 있고 다시 supernode사이의 상호 연결을 위하여 연결구조가 있다.

3) Supernode

병렬처리 시스템에서 통신은 중요한 역할을 하는데,[18] 기존의 프로세서에서 통신 병목 현상이 프로세서와 메모리 사이에 있었던 것처럼 프로세서 사이의 통신에서도 제약이 따르고 아무리 완전히 스위치 되는 시스템을 사용하여도 이 현상은 없어지지 않는다. 또한 정적인 네트워크를 사용한다면 시스템 내의 다른 프로세서와 공유하는 분할된 자료를 처리할 때 많은 경우에 통신의 복잡도가 알고리듬의 복잡도보다 더 심하게 된다. 따라서 프로세서간을 연결할때 분할된 자료 구조사이의 연결을 반영하여 많은 부분에서 국부적인 통신으로 문제 해결을 할 수 있도록 해야만 병렬 처리의 효과를 극대화시킬 수 있다. 처리소자 네개의 통신용 링크를 가진 경우에 직접 구현될 수 있는 네트워크에는 2차원 mesh, a perfect shuffle exchange network, a butterfly network, 이 외에 다른 four-connected topology들이 있다. 하지만 어떤 경우에도 자료 분할이 주어진 네트워크에 맞지 않는다면 통신폭은 그 네트워크의 지름에 비례하고 그 시스템의 처리 속도에 반비례하여 감소된다. 결국, 통신에 의해 제한되는 응용에서는 추가된 프로세서에 대해서 선형적인 성능 증가 효과를 얻을 수 없다. 이러한 문제를 해결하는 한 방법으로 프로세서 사이에 크로스바 스위치나 그에 상당하는 것을 이용하여 임의의 연결이 가능하도록 하는것이 있다. 이러한 방법은 비록 많은 장점을 가지고 임의의 병렬성 정도를 가능하게 하지만 그런 종류의 스위치를 만드는 비용이 시스템에 있는 프로세서의 수의 제곱에 따라 증가한다. 반면 고정된 네트워크를 위한 비용은 추가된 프로세서의 수에 따라 선형적으로 증가한다. 하지만 high bandwidth serial circuit를 통해 통신하는 트랜스퓨터의 경우에는 임의의 연결이 가능하도록 하는 것이 비싸지 않다. 이에 트랜스퓨터를 이용하여 임의의 네트워크를 구성할 수 있도록 허용하는 시스템을 구성하기 위한 첫 시도로 제안된 것이 supernode 구조이다.[3] 그림4는 그 구조를 나타내며 이것은 영국의 ESPRIT 연구과제에서 처음 소개되었다.

구현된 병렬 컴퓨터는 이러한 supernode 구조를

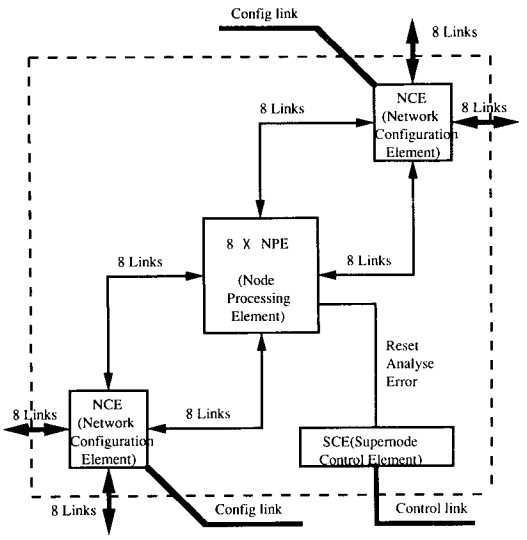

그림 4. 설계된 Supernode의 블록다이아그램
Fig. 4. Block diagram of supermode.

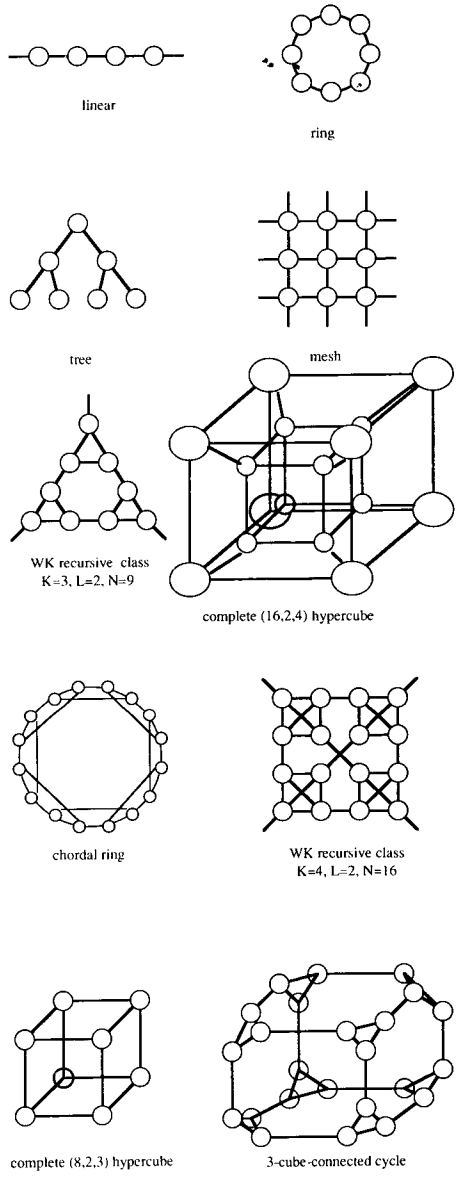

그림 5. PAKAC을 이용하여 구성할 수 있는 연결망의 종류
Fig. 5. Possible networks configured by KAPAC.

가지면서 전체 시스템을 중앙 집권적으로 관리할 수 있도록 설계하였다. 즉, 모든 NCE와 모든 PC 들을 제어하기 위하여 CPE(control processing element)를 따로 두었다. 이렇게 함으로써 전체 시스템을 효율적으로 관리할 수 있게 되고 따라서 이를 운영하는 운영 체제의 개발을 용이하게 할 수 있다. 또한 시스템에 있는 제어 대상들을 위해 따로따로 제어기를 두는 낭비를 줄일 수 있다.

4) 연결구조

Supernode 구조로 clustering 된 시스템을 잘 확장하여 좀 더 큰 시스템을 구성하기 위하여 연결 구조를 이층으로 구성하였는데, 하층에는 supernode 구조 내에 연결구조가 있고 상층에는 supernode사이의 상호 연결을 위한 연결구조가 있다. 이때에 구성되는 연결 구조는 supernode 사이의 상호 연결, 궁극적으로는 각 PE사이의 상호 연결을 항상 보장하도록 설계되었다. 그 구조는 하층을 이루고 있는 각 super-de의 NCE와 상층 연결 구조의 NCE의 상호 연결 해되어야 한다. 이때 사용된 NCE는 통신용 연결시켜주는 완전한 순열 스위치(a full permu..ion switch)이다. 설계된 연결구조 하에서 시스템내에 각 PE들은 적어도 하나의 NCE를 통하고 많아야 세개의 NCE를 통하여 다른 PE와 상호 연결된다.

여기서 만들어진 전체 시스템을 이용하여 직접 구성할 수 있는 정적인 네트워크는 그림5에 있는 것처럼 linear array, ring, chordal ring, tree, nearest neighbor mesh, 3-cube, hyper-cube, 3-cube-connected cycle, WK-recursive class에서 K가 4보다 작거나 같고 N이 32보다 작거나 같은 경우, hypertree, multitree 등으로 각 통신 노드에서 네개의 통신 링크까지 사용하는 경우들이다.[19,20,21,22] 또한 앞의 조건을 만족하면 문제 해결을 위해 특별한 네트워크도 구성 할 수 있다. 이러한 연결구조에 의하여, 응용프로그램에

맞추어 네트워크를 적절히 구성할 수 있으며 이러한 연결구조의 장점은 재구성 특성을 갖고 있는 다른 시스템[34,35] 들과는 다르게 처리소자간에 고정된 링크가 존재하지 않으므로 즉 처리소자에 있는 4 개의 모든 링크들이 연결구조 즉 NCE들에 모두 연결되어 있으므로 처리소자간의 연결을 좀더 융통성 있게 해준다는 점이며, 단점으로는 크로스바 스위치 갯수의 증가로 인하여 연결구조를 구성하는 비용이 증가한다는 점이다. 이 연결구조를 제어하는 적절한 네트워크를 구성하기 위해서는 switch configuration program[32,33]이 필요하며 이는 3.5에서 자세히 설명할것이다.

5) PE (processing element)

PPU에서 사용되는 PE들은 각각 목적에 따라 RPE(root processing element), CPE(control processing element), NPE(node processing element) 들로 구분할 수 있다. 지금까지 주로 언급되었던 PE는 NPE인데 supernode 구조로 clustering 되어 있으며 이들 각각의 기능은 다음과 같다.

(1) RPE(root processing element)

RPE는 한개의 T800 트랜스퓨터와 4M byte DRAM으로 구성되어 있는데, 네개의 통신 링크중 한개는 호스트와 연결되어 있고 다른 한개는 CPE와 연결되어 있으며 나머지 두개는 DSE(distributor switch element)의 data 링크에 연결되어있어 임의의 NPE와 통신하기 위해서 사용된다. RPE는 PPU의 창구로서 HM과 PPU내의 각 PE사이의 통신을 중계한다. 또한 병렬처리를 위한 실행 파일을 만드는 것처럼 많은 메모리를 요구하는 경우에 사용된다.

(2) CPE(control processing element)

CPE는 한개의 T800 트랜스퓨터와 128Kbyte 또는 512Kbyte SRAM으로 구성되어 있는데, 네개의 통신 링크중 한개는 RPE와 직접 연결되어 있고 다른 한개는 DSE의 configuration 링크에 연결되어 있으며 나머지 두개는 DSE의 data 링크에 연결되어 있어 임의의 PE, NCE와 통신하기 위해서 사용된다.

CPE는 DSE를 조정하여 RPE나 CPE가 원하는 NPE, NCE의 configuration 링크, SCE에 연결되도록 한다. 또한 switch configuration program에 의하여 임의의 연결이 되도록 NCE들을 조정하는 일을 한다.

(3) NPE(node processing element)

NPE는 한개의 T800 트랜스퓨터와 128Kbyte 또는 512Kbyte SRAM으로 구성되어 있는데, 네개의 통신 링크가 NCE의 data 링크에 연결되어 시스템 내에 있는 다른 NPE와 함께 임의의 연결 구조로 구성될 수 있다. NPE는 supernode내에 PE로서 시스템내에 다른 NPE와 함께 병렬처리를 위해서 사용된다.

Ⅲ. 소프트웨어

1. 개요

KAPAC에서의 소프트웨어를 크게 두가지로 나눠보면 HM을 위한 것과 CM을 위한 것이 있다. HM을 위한것은 전체 시스템을 운영하는데 필요한 기본적인 것(UNIX system V)과 이에 관련되면서 부가적인 device driver, 그 상에서 실행되는 file server가 있고, CM을 위한 것은 프로그램 개발을 하기 위한 compiler, linker, configurer등의 tool과 자체 개발된 switch configuration utility, 이외에 일반 응용 프로그램들이 있다. KAPAC와 관련된 소프트웨어의 전체적인 구조를 도시하면 그림6과 같다.

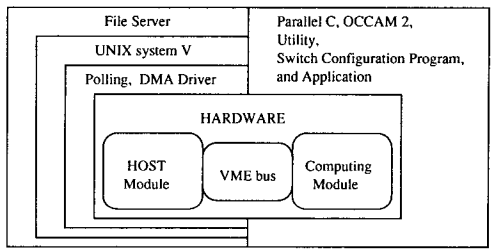

그림 6. KAPAC와 관련된 소프트웨어의 전체적인 구조

Fig. 6. Software block diagram of KAPAC.

2. 소프트웨어 개발환경

프로그램의 편집은 HM에서 기존에 사용하던 방법들을 이용하여야 하고 compile, link, configure등은 CM의 RPE에서 한다. 현재 사용한 병렬 프로그래밍 언어로는 OCCAM과 Parallel C가 있으며 그외 Parallel Pascal, Parallel Fortran등도 사용할 수 있다.[23,24,25,26]

3. Device Driver

HM에서 CPU보드는 MC68030 CPU를 이용한 MVME 147을 사용했고 운영 체제로는 UNIX system V가 사용되는데 hardware에서 access를 허용하기 위해 UNIX kernel 안에 device driver를 작성하여 두었다. 이러한 driver가 있으므로 해서 file server가 hardware에 대한 단순한 모델로 CM과의 인터페이스를 할 수 있게 된다. CM를 위하여 필요한 device driver는 polling driver와 DMA driver가 있다.

1) Polling Driver

적은 양의 data 또는 control data를 전송할 때 사용된다. 이 driver를 사용하는 경우는 HM의 컴퓨터가 실제 data를 전송할 책임을 갖고 있다. 이때 CM과의 handshake를 위해 상태검사를 하는 방법으로 직접 HM의 컴퓨터가 polling하는 방법과 CM로 하여금 interrupt를 하도록 하는 방법이 있다.

2) DMA driver

많은양의 data 또는 bootable code를 전송할 때 사용된다. 실제 전송은 DMAC가 담당한다. 전송이 끝난 후 DMAC의 interrupt 신호를 받아서 HM의 computer가 마무리 한다.

4. File Server

File server는 호스트 운영 체제상에서 실행되면서 CM에서 실행될 파일을 downloading하고 수행중인 프로그램이 호스트 파일 시스템과 그 외 다른 facility를 이용할 수 있도록 file process기능을 제공한다. Filer process는 사용자가 file을 create, open, read 및 write할 수 있도록 하는 간단한 filing system의 모델을 제공한다. Filer process는 두개의 channel을 통해서 사용자 프로그램과 정보 교환을 하는데, 한 channel은 사용자 프로그램이 file process에게 명령어 및 data를 보내기 위해서 사용되고 다른 하나는 filer process로 부터 사용자 프로그램으로 결과나 data를 전송하기 위해서 사용된다. 주어진 많은 utility들이나 tool은 file server portocol을 기반으로 해서 만들어 진다.

5. Switch Configuration Program

CM에서 프로그램을 수행시키기 전에 처리소자들간에 필요한 연결망의 구축이 선행되어야 한다. 이러한 연결망 구성은 CPE가 연결구조내의 NCE들을 알맞게 조정함으로써 이루어진다. 프로그램 개발시 사용자는 실제로 각 처리소자에 일을 할당하고 그들간의 연결망구성을 위해서 최소한 CM내부의 개략적인 구성을 이해해야 한다.

CM을 사용하는 사용자마다 이것의 개략적인 구성을 이해해야 하는 것은 상당히 불편한 일이므로, 이를 위하여 최소한의 지식만을 가지고 열결망을 구성할 수 있는 utility가 있어야 한다.

이를 위하여 switch configuration program을 개발하였으며 이것을 이용하면 사용자는 CM에 32개의 처리소자가 있고 각 처리소자는 4개의 링크만을 가지고 있다는 사실만을 가지고 프로그램을 개발할 수 있다. 즉 사용자가 정의한 가상 처리 소자들과 그들간의 연결을 가지고 이 프로그램은 실제 처리소자로의 매핑과 그들사이의 연결을 위하여 연결구조내의 NCE들을 제어해 준다. 이 방식의 장점은 위에서 설명한것 외에도 처리소자중 몇개가 고장이 났을 때에도 이 utility는 고장난 처리소자를 제외한 정상적인 처리소자로 매핑해주므로 사용중인 모든 프로그램을 고칠 필요가 없다는 점이다. 즉 job 할당을 이원화 시켜서 사용자가 프로그램을 개발시는 각 처리소자가 4개의 링크를 가지고 있다는 사실만을 가지고 가상적으로 처리소자간 연결망을 구성하고 실제 매핑은 프로그램 수행시할 수 있도록 하였다.

그림7은 처리소자가 7개 필요하고 이들의 연결이 나무(tree) 구조이어야 하는 병렬프로그램의 경우 switch configuration program에서의 입력예를 나타낸 것이다. 이 그림에서 첫번째와 세번째열은 처리소자의 번호이며 두번째와 네번째열은 처리소자내의 링크번호이다. 즉 각행은 두개의 처리소자 사이의 연결상태를 나타내며 6개의 행은 모든 연결상태를 서술하고 있다. Switch configuration program은 이것을 입력으로 현재 사용 가능한 처리소자 중에 7개를 선택해서 이를 실제로 연결시켜 준다.

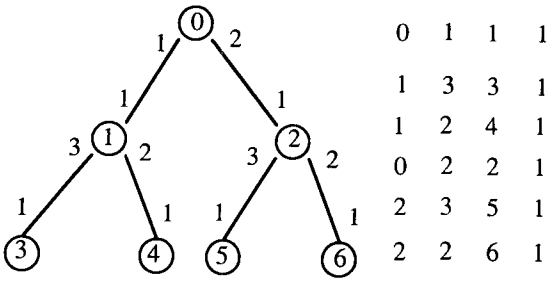

그림 7. Switch Configuration program의 입력 예
Fig. 7. Input example to switch configuration program.

6. 응용 프로그램 예

본 논문에서는 임의의 연결이 가능함을 보이기 위해 몇가지의 네트워크에 맞는 응용 프로그램을 소개한다. 먼저, 가장 단순한 선형 연결의 경우[27]에 Pi의 근사값을 계산하는 문제를 해결하기 위한 프로그램을 작성하였다.[28] 이 문제는 여러개의 동일한 구간으로 나눈 후 각 구간에서의 면적의 합을 구하므로써 얻을 수 있다. 따라서 많은 통신량이 필요없고, 처음 초기화 단계와 합산하는 단계에서만 통신이 필요하다. 둘째, 링(ring) 연결의 경우에는 영상에 대한 창

연산(window operation)하는 문제에 적용한다.[31] 영상을 수평 분리하여 링으로 연결된 각 PE에 할당하므로써 통신량을 줄이면서 병렬 처리할 수 있게 한다. 셋째, 나무(tree) 연결의 경우에는 sorting이나 searching에 적당하므로 quick-sort 프로그램을 작성하였다.[29,30] 넷째, 메쉬(mesh) 연결의 경우에는 2차원 편미분 방정식을 해결하기 위해 적당하다.[10] 즉, 2차원 자료를 수평 및 수직으로 분할하여 할당하므로써 자료구조와 PE 연결 구조가 잘 상응하도록 할 수 있고 결과적으로 보다 낮은 병렬처리 효과를 얻을 수 있게 한다. 다섯째, complete(8, 2, 3) hypercube의 경우에는 K-means 알고리듬에 의한 optimal level clustering을 병렬화하여 수행시켜 보았다.[36] Clustering을 병렬화 할 경우 모든 프로세서간의 효율적인 통신이 필요하기 때문에 hypercube topology가 적절하다. 앞에서 언급한 것들을 그림8에 나타내었다.

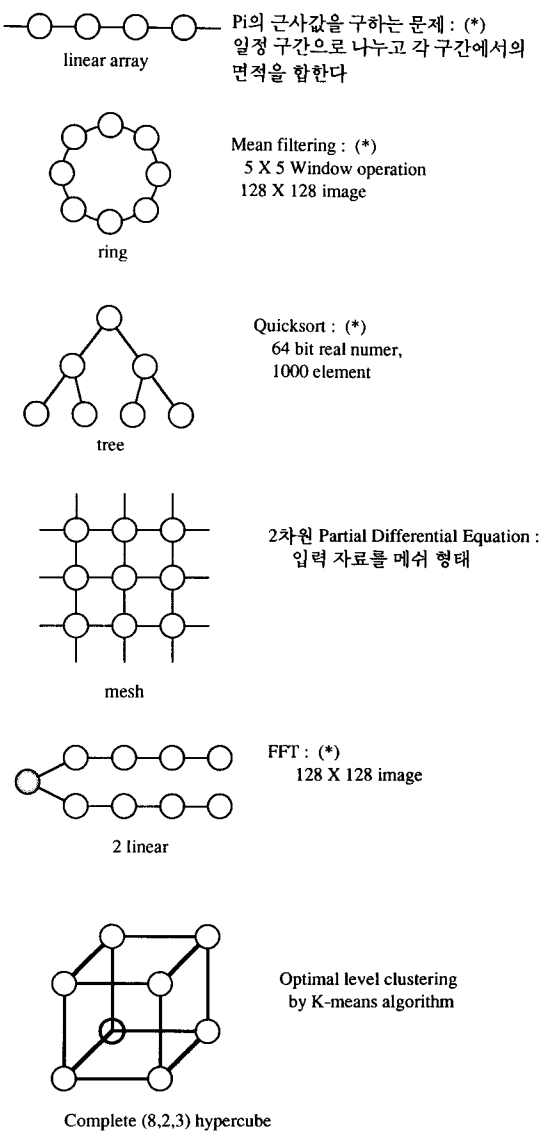

그림 8. 응용 프로그램 예
Fig. 8. Application examples.

Ⅵ. 성능평가(Performance Evaluation)

1. 성능평가 방법

개발된 병렬컴퓨터 KAPAC의 성능을 평가하기 위하여 다수의 응용프로그램을 병렬화하여 수행해 보았으며, 그 결과를 다른 컴퓨터 시스템과 비교하여 보았다. 이 병렬화된 프로그램의 병렬처리 효율을 평가하기 위한 척도로써 다음의 요소들을 사용하였다.[5,7]

(1) 수행시간(running time)

병렬처리 효과를 평가하는 가장 중요한 척도로서 실행을 시작해서 끝날때까지의 경과된 시간으로 정의된다. 수행시간은 경로이동 구간(routing steps)과 계산 구간(computational steps)으로 구성되며, 경로이동시간은 자료가 네트워크를 타고 처리기 사이를 이동하는데 걸리는 시간이며 계산시간은 한 처리기 내에서 자료에 가해진 연산 작업에 소요된 시간이다.

(2) 가속성(speed up)

크기 N의 문제에 대하여 직렬 대 병렬시간의 비를 가속성이라고 정의하고, 처리기의 수가 P라 할때 궁극적인 목표는 S(P)=P이며 일반적으로는 P보다 작다.

(3)비용(cost)

수행 시간과 처리기수의 곱을 비용이라 정의한다. 이는 주어진 문제를 풀기위해 사용된 비용을 의미한다.

(4)효율성(efficiency)

계산에 소요되는 비용효율의 척도로서 직렬 비용 대 병렬 비용의 비로 정의되며, 가속성이 사용한 프로세서 수에 접근 할수록 효율성은 높아진다.

2. 수행시간 측정

트랜스퓨터의 타이머는 그 프로세스의 우선 순위(high/low priority)에 따라 1 microsecond 또는 64microsecond 마다 증가된다. 그리하여 high priority process의 경우 1시간 10분 정도의 주기로 사이클을 돌며, low priority process의 경우 76시간 정도의 주기로 도는 시간값을 얻을 수 있다.[11] 그러므로 이를

표 1. 수행시간의 측정 및 평가
Table 1. Running-time and pseed-up factor of various examples.

Problem (topology)	NProc	Time	Speed up	Cost	Efficiency
Pi (linear)	1	19.450752	1.000000	19.450752	100.00
	2	9.725440	1.999987	19.450880	100.00
	8	2.431552	7.999316	19.452416	99.99
	16	1.216128	15.994000	19.458048	99.96
Mean Filtering (ring)	1	2.152378	1.000000	2.152378	100.00
	2	1.105402	1.947145	2.210804	97.36
	8	0.310685	6.927847	2.485480	86.60
	16	0.178089	12.085968	2.849424	75.54

이용하여 병렬 프로그램의 수행시간 측정을 용이하게할 수 있다.

앞에서 언급한 몇가지 응용 프로그램중 Pi값 계산과 Filtering에 대해 수행 시간의 측정 및 효율성을 표1에 나타냈으며 처리소자 수에 따른 가속성 그래프를 그림9에 도시하였다.

이 결과로 부터 Pi값 계산은 처리소자간 통신량이 전체 계산과정에 미치는 영향이 적은 task이므로 처리소자의 수만큼 거의 속도향상을 얻었음을 알 수 있다. Mean Filtering의 경우에는 처리소자간 통신시간이 차지하는 비율이 전체계산 과정에 미치는 영향이 Pi값 계산보다 상당히 많으므로 사용한 처리소자의 수만큼 속도향상을 얻지 못했음을 알 수 있다.

표2는 Pi의 근사값을 구하는 문제에 대해서 다른 컴퓨터 시스템에서 수행시켰을 때의 계산 시간과 KAPAC에서의 계산시간을 비교한 것이다. KAPAC에서 하나의 처리소자를 사용한 경우 소요시간이 거

표 2. Pi값 계산에 있어 다른 컴퓨터들과의 비교
Table 2. Comparison with KAPAC and other computers on Pi calculation.

Computer	Processor	Number Processor	Escaped Time (second)
Sun 3/110	MC 68020 MC 68881	1 1	112.633
Sun 4/sparc 1	Sparc	1	14.683
Solbourne	Sparc	2	5.766
KAPAC	T800	1	19.450
KAPAC	T800	2	9.725
KAPAC	T800	8	2.431
KAPAC	T800	16	1.216

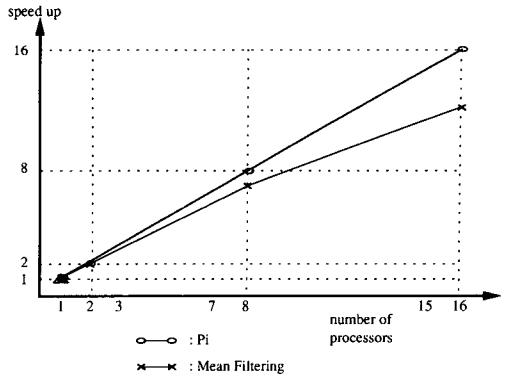

그림 9. 프로세서의 수에 따른 가속성 그래프
Fig. 9. Speedup graph.

의 SUN4/sparc 1에서의 소요시간의 약 2배가 되었으나 사용처리소자의 갯수를 늘여감에 따라 소요시간이 훨씬 줄어듬을 알 수 있다.

V. 결 론

본 논문을 통하여 제시된 KAPAC(KAist PArallel Computer)는 처리소자 사이의 네트워크 구성을 소프트웨어로 조정할 수 있도록 하였다. 그리고 마스터 부분과 슬레이브 부분으로 이루어지는데, 슬레이브 부분은 필요에 따라 계산 능력을 조정할 수 있도록 8개의 처리 소자를 갖고 있는 네개의 슈퍼 노드(supernode)와 연결구조로 구성되어 졌다. 또한 32개의 처리소자 각각에 대해 제어 및 오류 발생 검사를 할 수 있게 하였다. 그리고 KAPAC내에 있는 상호 연결망을 구성하는 많은 스위치 소자와 각 처리소자를 감시하는 제어소자를 중앙 집권적으로 조정할 수 있게 했다. 즉, 본 논문에서 제시된 KAPAC는 reconfigurability, partitionability, modularity, centralized control등의 특징을 갖는다.

본 논문에서 소개한 병렬 컴퓨터는 복잡하거나 많은 계산이 요구되는 일을 병렬로 처리하여 속도 향상을 시킴으로써 실시간 처리 및 고성능 처리를 필요로 하는 많은 응용 분야에 대한 다목적 계산 능력을 제공하게 된다.

또한 병렬처리 하드웨어 및 시스템 개발 과정에서 획득한 기술력은 대규모 병렬처리 컴퓨터의 밑거름이 될 것이고, 병렬처리 응용 프로그램 개발 환경을 제공하므로써 많은 응용분야에서 병렬처리를 도입할 수 있는 계기를 마련하였다.

지금 개발된 KAPAC의 보다 나은 활용을 위한 automatic switch configuration program 개발과, multitasking, efficient task allocation을 위해 필요한 kernel 개발에 대해서는 추후 과제로 남겨둔다. 또한 KAPAC를 이용하는 많은 응용 프로그램을 작성하므로써 새로운 문제가 인식되고 개선되기를 기대한다.

參 考 文 獻

[1] K. Hwang and F.A. Briggs, Computer Architecture and Parallel Processing McGraw-Hill, 1984

[2] K.Hwang. "Multiprocessor supercomputers for scientific/engineering applications," *IEEE Computer*. June. 1985

[3] R.W. Hockney, and C.R. Jesshope, Parallel Computers 2,Adam Hilger, 1988.

[4] M.J.Flynn. " some computer organizations and their effectiveness," *IEEE Transactions on computer,* pp.948-960. Sept. 1972.

[5] D.I. Molodovan, Moden Parallel Processing. univ. of southern California. Jan. 1986.

[6] A.H.Karp."Programming For parallelism" *IEEE Computer* pp.43-57 May.1987

[7] D.Tabak. Multiprocessors, Prentice Hall. 1990.

[8] L.N. Bhuyan. Q.Yang.and D.Pgrawal. "Performance of multiprocessor interconnection networks." *IEEE Computer* pp. 25-37 Feb. 1989

[9] R.Duncan. "A survey of parallel computer architectures." *IEEE Computer* pp5-16 Feb. 1990

[10] D.A.Reed. and R.M. Fujimoto. Multicomputer Network. The MIT Press. 1987

[11] C.L. Seitz. "The Cosmic Cube." Comm. ACM. pp. 22-23, Jan. 1985.

[12] W.C.Athas. and C.L.Seitz. "Multicomputers: message-passing concurrent computers." *IEEE Computer* pp. 9-24 Aug. 1988

[13] C.A.R. Hoare. "Communication sequential processes." Comm.ACM pp. 666-677 Aug. 1978

[14] C.A.R.Hoare. Communication sequential processes. Prentice-Hall International. 1985

[15] Inmos Limited. The Transpsuter Databook. Inmos. 1989.

[16] Motorola Inc. Cirect Memory Access Controller(DMAC). Motoeola. 1986

[17] Motorola Inc., VME bus Specification Manual. Motorola, 1985.

[18] W.H.Burkhardt, "Limitations to parallel processing," Ninth Annual International Phoenix Conference on Computers and Communication. pp. 86-93, 1990.

[19] Tse-Yun Feng, "A Survey of Interconnection Networks," *IEEE Computer*, pp. 12-27, Dec. 1981.

[20] D.A.Reed, and D.C.Grunwald, "The performance of multicomputer interconnection networks." IEEE Computer, pp. 63-73, June 1987.

[21] E.chiricozzi, and A.Damicoo, Parallel Processing and Applications, North Holland, 1988.

[22] D.P.Agrawal, and V.K.Janakiram, "Evaluationg the performance of multicomputer configurations," *IEEE Computer,* May. 1986.

[23] D.Pountain, and K.May, Tutorial Introduction to Occam Programming, BSP Progcssional Books, 1988.

[24] Inmos Limited, Occam 2 Rcference Manual, Prenticc-Hall International, 1988.

[25] Inmos Limited, Occam 2 toolsct Uscr Manual, Inmos, Apr. 1989.

[26] Inmos Limited, 3L Parallel C IMS D711 Delivery Manual, Inmos, Feb. 1989.

[27] D.C.Mason, "Linear quadtree algorithms for transputer array," *IEEE Proceedings,* pp. 114-128, Jan.1990.

[28] R.G.Babb II, Programming Parallel Processors, Addision-Wesley, 1988.

[29] E.Horowitz, and s.Sahni, Fundamentals of Computer Algorithms, Computer Science Press, 1978.

[30] 최승욱, 트랜스퓨터를 이용한 병렬 계산 가속기의 설계 및 구현, KAIST석사학위 논문, 1990.

[31] 강휘삼, 공장 자동화용 컴퓨터시스템을 위한 병렬 컴퓨터의 설계 및 구현, KAIST석사학위 논문, 1991.

[32] M.Tudruj and M. Thor, "The architecture of a multilayer dynamically reconfigurable transputer system," International Conference on Parallel Processing, 1991.

[33] L.Jin and L.Yang and C.Fullmer and B.Olson, "Dynamic reconfigurable architecture of a Transputer-based multicomputer system," International conference on Parallel Processing, 1991.

[34] Inmos Limited, "IMS BOO8 IBM PC Module, " Inmos, 1988.
[35] 허남칠, 정창성, 오종훈, 이재용, 이전영, 방승양, "병렬 컴퓨터 POPA," 정보과학회지, 제6권 제6호, p31-p38, 1988.
[36] 성동수, 박규호, "Parallel algorithm for level clustering of patterns on Transputer based machine," KAIST Internal Report. 1991.

著 者 紹 介

成 桐 洙(正會員)

1987年 한양대학교 공과대학 전자공학과 졸업(학사). 1989年 한국과학기술원 전기 및 전자공학과 졸업(석사). 1989年~현재 한국과학기술원 전기 및 전자공학과 박사과정. 주관심분야는 병렬처리, 인공지능, 컴퓨터비젼, 지능컴퓨터 등임.

姜 輝 三(正會員)

1989年 한양대학교 공과대학 전자공학과 졸업(학사). 1991年 한국과학기술원 전기 및 전자공학과 졸업(석사). 1991~현재 삼성전자 가전연구소 영상연구실 연구원. 주관심분야는 병렬처리, 영상처리 등임.

崔 勝 旭(正會員)

1988年 2月 한양대학교 전기공학과(학사). 1990年 2月 한국과학기술원 전기 및 전자공학과(석사). 1990年 3月~1991年 6月 한국통신 연구개발단 교환연구부 전임연구원. 1991年 6月~현재 한국통신 서울전자교환 운용연구단 교환연구국 전임연구원. 주관심분야는 지능 컴퓨터, 병렬처리, B-ISDN 등임.

朴 圭 晧 (正會員) 第27號 第9號 參照
한국과학기술원 전기 및 전자공학과 교수