QÜESTIIÓ, Vol 11, n° 3, pp. 73-86, 1987

MULTISTAGE INTERCONNECTION NETWORKS IN MULTIPROCESSOR SYSTEMS. A SIMULATION STUDY.

VICTOR LOPEZ DE BUEN

Facultat d'Informàtica de Barcelona

The principal modelling and simulation features of multistage interconnection networks operating in packet switching are discused in this paper. The networks studied interconnect processors and memory modules in multiprocessor systems. Several methods are included to increase the bandwidth achievable with this kind of networks. Besides usign network buffering, the possibility of having queues of requests at the memory modules is considered. Network conflicts can be reduced using a second network to return requests from the memory modules. The connection of more than one processor or memory module to each of the multistage network input or output lines allows the interconnection of large multiprocessor systems using small multistage networks. This is implemented using a single shared bus connection. The effective bandwidth of these networks is compared to that of circuit switching multistage networks and Crossbar. Simulation results reflect an important improvement in network perfomance.

Keywords: Interconnection networks, perfomance analysis, packet communication, multiprocessor systems, simulation.

1. INTRODUCTION

Multistage interconnection networks have been widely proposed for multiprocessor systems, specially in SIMD architectures [15,16]. Digit–controlled type are the most commonly used. Processor requests are routed through this networks according to certain digit (bit) of the destination address at each stage of the network. This provides a very·simple request routing that allows local control at each switch with no need of external or global control in the network.

–V. López de Buen - Facultat d'Informàtica de Barcelona - Pau Gargallo, 5 - Barcelona.
–Article rebut el setembre de 1986.

Moreover, the low cost of network components gives a better perfomance per cost than Crossbar as the number of processors and memory modules increases [13]. This class of multistage networks are also known as Delta networks [13,1] or as Shuffle/Exchange networks [6,12], and includes several networks that have been proposed as special cases. Some of them are: Omega [7], Indirect Binary n–cube [14], Cube [15] and Baseline [19].

Packet communication improves the bandwidth achievable with multistage networks [1,5]. It is obtained including queues (buffers) of requests in the basic switching modules at each stage of the network. Computers designed under this concept are HEP [17] and NYU [3]. Two more methods to increase network bandwidth are suggested in this paper: the blocking of requests in the last stage is reduced including queues at each memory module, and a second network is used to route back the requests from memory modules to processors to avoid the conflicts among going and returning requests.

Packet switching also permits the connection of several processors to the multistage network input lines or several memory modules to each of the output lines [9]. This provides the advantage of connecting a large number of processors and memory modules using a quite small interconnection network. The loss in effective bandwidth is not too large and in some cases bandwidth increases with this configurations.

The complexity of this kind of systems makes very difficult the analytical analysis, so simulation becomes an indispensable tool. A simulator has been developed to study all of these interconnection alternatives. The simulator and some preliminary results have been presented before in [8]. In that paper smaller multiprocessor systems are studied and only the two smallest basic switching modules are considered. The parametric design of the simulator permits a very fickle analysis of this family of networks by changing the number of processors or memory modules, interstage interconnection patterns, basic switching elements size, number of buffers, loading conditions, statistical evaluation methods, etc. In the last section of this paper an analysis of simulation results for the perfomance of different networks is presented.

2. NETWORK MODELLING.

Multistage interconnection networks are constructed from cascaded stages to divide the task of permutation in the network into several sub–tasks of lower complexity. Each stage consists of a set of basic switching modules that represent itselves a small crossbar network. The outputs of the switches of each stage are routed to the inputs of the next stage switches through some interstage wiring pattern that defines the network topology (Omega [7], Indirect binary n–cube [14], etc.).

This kind of networks are termed digit–controlled since each switching module is controlled by a single digit from the destination address. In practice, the base of the digits required for module control must be a power of 2 and the size of the modules cannot be very large due to cost and technological limitations [13]. This suggested the implementation of 2×2, 4×4 and 8×8 modules as the only three possible sizes in the network. These switches can connect any of the input lines to any of the output lines with the restriction that two or more input lines cannot be connected to the same output line at the same time. When two or more input lines have requests that attempt to pass through the module to the same output line a conflict is produced. In this case, one of the requests is equiprobably selected and passed while the others wait to the next cycle. The principal characteristics of these networks are:

- The network has N input and output lines and there is an unique path from each input line to each output line.
- All basic switching modules are identical and its size is $b \times b$. The value of b can be 2, 4 or 8.
- Each stage in the network has N/b switching modules.
- The number of stages in the network is $Log_b N$, so it is not possible to connect N input lines in the network with N output lines when N is not a power of b.

The switching modules can store a fixed number of waiting request in queues at their input lines so the network is able to work in a packet communication environment. Packet communication reduces the blocking probability in the network and permits the connection of more than one processor to each of the multistage network input lines or more than one memory module to each of the output lines. This can be implemented using a single shared bus connection [18], and allows working with a great number of processors and memory modules interconnected with a relative small multistage network. Shown in figure 1 is an interconnection general diagram of multistage networks with 2×2 switching modules. The number of input and output lines in the network is N and it can be connected to each input line X processors and to each output line a number Y of memory modules. So we can connect this way $N \times X$ processors and $N \times Y$ memory modules with a multistage network of Z stages. Z equals $Log_2 N$.

To denote a multiprocessor system with this characteristics we can use a triplet: processors/network/memory modules, based on that described in [18]. The general representation of the configuration of the system is $X/N \times M$ NETb/Y. X, N and Y are shown in figure 1 and have been described before. M represents the number of network output lines, which equals N (balanced networks) in all the networks analysed in this paper. NET is the multistage network configuration used and b is the size of network switching modules ($b \times b$).

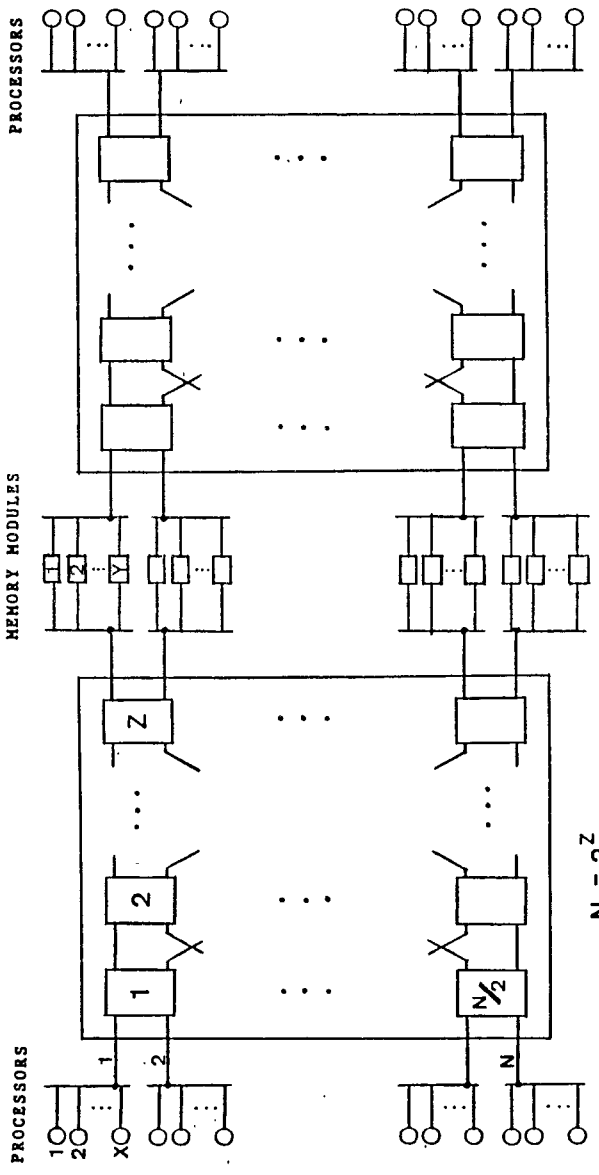

Fig.1. General diagram of interconnection using multistage networks with 2X2 switching modules.

FIGURE 1. General diagram of interconnection using multistage networks with 2×2 switching modules.

As an example, consider a system with 16 processors and 32 memory modules. If an 8×8 indirect binary n–cube network [14] is used (in this case, we can only use 2×2 network modules) we have: $2/8 \times 8$ INDCUBE2/4.

It is necessary to establish a set of assumptions about the operation of the networks in order to model the system:

- Requests are information packets which contain both the data to be transfered and the labels of both the memory module to which the data are to be passed and the originating processor.
- Each processor generates random and independent requests uniformly distributed over all memory modules. Every time a processor receives a returning request sends a new one in the next cycle (or with a probability $p < 1$, which represents the internal processing time). Once a processor has sent a request it is not able to send another until the last one is back. If a request cannot be accepted in any cycle, the processor will be blocked and will keep the same request for sending in the next cycle or succeeding cycles, until served.
- The time unit in the system is the switching cycle which can be either in a network module or in a single bus connection.
- Each request needs only one access to a single memory module. The service time of a request in a memory module is a constant value and a multiple of the switching cycle. Memory modules can admit (or not) requests stored in queues (buffers) when this requests are blocked at the input of the memory module (memory module is busy) or at the output of the module when the network cannot accept the returning request in that moment.
- When a conflict is produced in an interstage connection in the network between requests going to and returning from the memory modules we can choose among two arbitration policies:
  - Memory modules priority. Priority is conceded to the requests that return from the memory modules.
  - Processors priority. In this case, pass priority is assigned to requests going to the memory modules.
- It is possible to use a second network for routing the requests that finish their stay time in memory back to their originating processors. This avoid the conflicts mentioned before increasing system efficiency.

3. THE SIMULATOR

The simulator is written in Fortran IV language and it has been designed under a modular concept that allows differentiating the diverse calcul stages

during the simulation. This facilitates further adaptations of interest. The simulation of interconnection networks has an special purpose in analysing different alternatives in the pass of the information through the network [10]. Accordingly, the simulator works around a set of paremeters whose value can be changed in consecutive simulations in order to make comparative analysis. There are two kind of parameters: to modify the behavior of the network and to control the simulation.

The set of the principal parameters used in simulation analysis can be separated in five groups to explain their specific functions. Most of these parameters have default values in the simulator that can be modified in successive simulations.

1) Network structure. There are several parameters for defining the structure of the network, like the number of processors and memory modules used in the system, the number of input and output lines of the network, the size of the basic switching modules, the interstage interconnection patterns. There are several configurations of multistage interconnection networks proposed by the simulator.

2) Network operation parameters. This parameters have a direct influence in the traffic of the requests through the network. The emission of new requests is a function of the internal processing, the return of the requests from memory modules is controlled by the service time, memory modules can accept (or not) requests in queues of variable size, and the number of buffers in the queues of the network switching modules can be changed.

3) Network analysis ways. There are three different ways to study the model of the network employed:

a) Requests go through the network going to the memory modules and returning from them. In this case a pass priority must be assigned to solve conflicts among requests going and returning.

b) Requests do not return through the network. Processors send new requests without waiting for previous ones.

c) Requests return from memory modules using a second network to avoid conflicts with the going requests.

4) Statistical evaluation parameters. The simulator incorporate several output analysis methods that are used according to the value of certain parameters. Simulation length, transient behavior and the precision in confidence intervals estimation are also controlled in a parametric way.

5) Parameters for the random number generation. The simulator provides six streams of random numbers that can be assignated to different parts of the model calculus, and proposes six seeds for these streams whose values can be changed by the user if wished.

4. SOME SIMULATION ASPECTS.

Three different methods for output data analysis were included in the simulator: Independent Replications, Batch Means and Spectrum analysis. The incorporation of the Regenerative method [4] was considered, but regeneration points are very difficult to detect in this kind of networks that generate a large number of different states. Independent replications method [2] consists in carrying out several independent simulations and obtaining average values among them. In batch means method [2] the simulation is divided in batches of the same size and average values are considered. Spectrum method estimates the effect of correlation among the elements of the sample. Spectrum analysis demostrated being the best method in an accuracy per simulation length estimation, so all the results presented here have been calculated under this method. The algorithm used is that suggested in [1]. The three methods assume that data collection begins at a point in which initial conditions no longer influence behavior (steady state), so it is very important to be sure when that point is reached during the simulation. The method used in the simulator to evaluate this transient state is based on the comparison of sets of near partial results considering both the magnitude and the variation of the differences.

The principal perfomance measure used in interconnection networks is that of expected bandwidth (EBW): the mean number of accesses to memory modules in a processor cycle. In the simulation it is evaluated as:

$$\text{EBW} = \text{ACC} \times \text{CYREQ} \;/\; \text{TOTCYC} \tag{1}$$

$$\text{CYREQ} = (\text{NST} \times 2) + \text{CYMEM} \tag{2}$$

ACC is the number of successful accesses, CYREQ is the minimum number of cycles for a request to complete its round trip through the system (processor cycle), and TOTCYC is the total number of cycles simulated. CYMEM is the average number of cycles a request stays in a memory module, and NST is the number of stages. NST equals the number of stages in the multistage network when there is only one processor connected to each of the network input lines; but when two or more processors are connected, NST is increased in one stage due to the single bus necessary to establish the connection. Expected bandwidth can also be calculated using the mean time (mean number of cycles) a request needs to travel through the network (mean request interarrival time). If TIREQ is this mean time and PRO the number of processors:

$$\text{EBW} = \text{PRO} \times \text{CYREQ} \;/\; \text{TIREQ} \tag{3}$$

Packet switching in multistage networks forces the use of a relative measure for comparison among this networks and single stage networks (or circuit switching multistage networks). Relative expected bandwidth is defined as:

$$\text{EBWr} = \text{EBW} \times (\text{CYMEM} + 2) \,/\text{CYREQ} \tag{4}$$

The value of CYREQ for a single stage network is of course CYMEM + 2.

5. SIMULATION RESULTS.

In this section various interconnection options for a 64×64 multiprocessor system (64 processors and 64 memory modules) are analysed. SIRI simulator [8] allows a wide variety of interconnection possibilities and the most representative of them are presented here. Relative expected bandwidth (EBWr) is shown in the following figures as a function of memory modules service time (CYMEM). This parameter has a direct influence in network bandwidth when the system is working in packet communication and it is independent respect circuit switching networks because in this case processor requests are synchronized. The constant values of bandwidth for Crossbar and circuit switching Omega network are included for comparison. The network configuration drawn is indicated in the figures using letters. The relation used is the following:

a)	8/ 8×8 OMEGA2	/8.	4 stages	$(3+1)$
b)	8/ 8×8 CROSSBAR	/8.	2 stages	$(1+1)$
c)	4/ 16×16 OMEGA2	/4.	5 stages	$(4+1)$
d)	4/ 16×16 OMEGA4	/4.	3 stages	$(2+1)$
e)	2/ 32×32 OMEGA2	/2.	6 stages	$(5+1)$
f)	1/ 64×64 OMEGA2	/1.	6 stages	$(6+0)$
g)	1/ 64×64 OMEGA4	/1.	3 stages	$(3+0)$
h)	1/ 64×64 OMEGA8	/1.	2 stages	$(2+0)$

Between parenthesis is indicated the number of stages in each interconnection system adding the stages of the multistage network plus the stage of the single bus (when needed) used to connect more than one processor to each input line.

Omega network [7] is used in this analysis due to its great acceptance [1,5,13,18], but simulation results using other networks (like indirect binary n–cube [14], Baseline [19], etc.) indicate that all of them has approximately the same perfomance in the environment considered. This agrees with mentioned in [1] and [13] about Delta networks. In fact, in [15] is shown that

Omega network and Indirect binary n–cube network can be made equivalent by an address transformation.

The results reported in this section were carried out varying the values of the following parameters:

1) The service time of the memory modules (average number of cycles a request must stay in memory).
2) The size of the network.
3) The size of network switching modules.
4) The queue size of the network modules.
5) The queue size of the memory modules.

Processors pass priority demostrated being the best alternative when a conflict is produced among two requests in the interstage connections (one going to the memory modules and another returning); so the results presented in this paper were obtained under this kind of priority.

Spectrum method was used for estimating confidence intervals of results and the accuracy considered for this intervals was 95%. The simulation lenght was evaluated as a function of minimum turn around time of requests. The results obtained this way were always around a confidence semi–interval of 1% of mean values.

Figure 2 shows the important improvement achieved in effective bandwidth (EBWr) in almost all the cases studied using this family of networks. This value increases as the service time of the memory modules (CYMEM) grows. A longer service time produces a reduction in blocking and conflicts among the requests when they go through the network.

$1/64 \times 64$ OMEGA2/1 network (f) is the packet switching version of the 64×64 Omega network operating in circuit switching used for comparison. Its perfomance when the number of cycles for service time is small is worst than that of circuit switching (like the smaller networks), but increase very fast as the number of cycles required grows. If we use 4×4 (g) or 8×8 switching modules (h) in the construction of the network we obtain a much better perfomance, specially for lower values of CYMEM. This is due to the reduction of conflicts at each stage using bigger modules and the reduction of stages, making a faster network. However, the growth in network complexity increases its cost.

Important results can also be obtained using smaller networks. If we connect two elements (processors and memory modules) to each connection line we can use a 32×32 network (e) that has a similar behavior with ligthly inferior results. We observe that improvement reduces when we connect 4 elements to each line, but simpler multistage networks are employed. The traffic in the

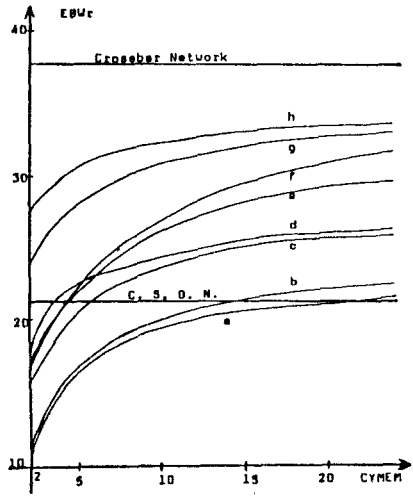

Fig.2. Relative Expected Bandwidth (EBWr) versus Service Time of Memory Modules (CYMEM).

FIGURE 2. Relative Expected Bandwidth (EBWr) versus Service Time of Memory Modules (CYMEM).

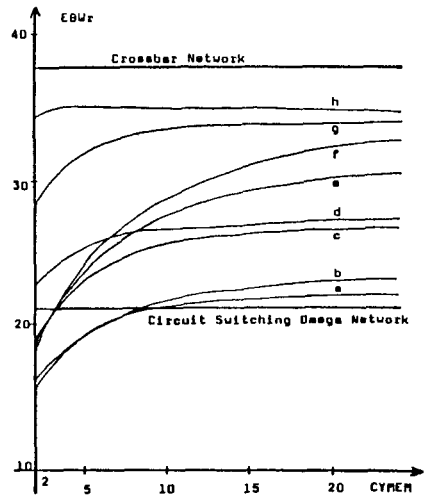

Fig.4. Connection using two networks.

FIGURE 4. Connection using two networks.

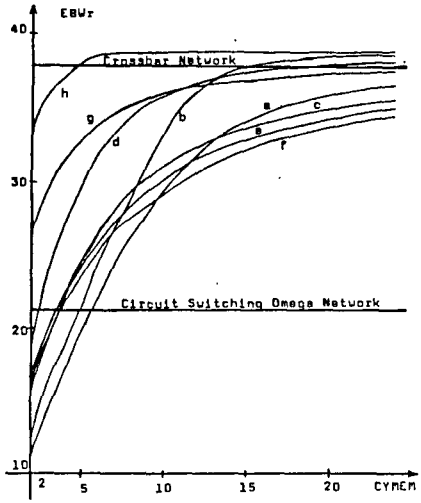

Fig.3. Including memory modules buffering.

FIGURE 3. Including memory modules buffering.

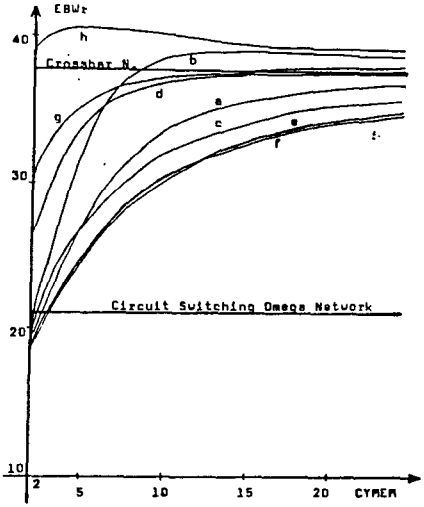

Fig.5. Including both memory buffering and connection with two networks.

FIGURE 5. Including both memory buffering and connection with two networks.

network is increased too much by input flow when the smallest networks are used connecting 8 elements to its input and output lines, so very poor results are obtained.

The same interconnection systems described before are shown in figure 3, but now requests can form queues at the memory modules. All of them improve their bandwidth, but now networks with less stages has a better perfomance. This is due to system efficiency is now a function of how fast the network can route requests to and from memory modules buffers, since no request has to wait in the network switches when memory modules are busy. This reduces of course the conflicts in the network.

If we interconnect the systems of figure 2 (with unbuffered memory modules) using a second network (identical) to route back the requests from the memory modules to the originating processors, we obtain the results shown in figure 4. This reduces network interferences avoiding conflicts among going and returning requests. Large values of the memory modules service time (CYMEM) also reduces network interferences, so the improvement in EBWr is much more important when CYMEM is small.

The combined effect of memory modules buffering and the use of a second network to route back requests (fig. 5) improves effective bandwidth in a considerable amount only for small values of CYMEM, and for networks with a less number of stages, where more conflicts are presented. Networks constructed with 8×8 switching modules (b, h) show a saturation process as the value of CYMEM grows, due to its size (only two stages) which allows a very fast transit through the network.

All the simulations described before assume infinite queues, but large queues are not required to produce those results. The four networks constructed with 2×2 switching modules (a, c, e, f) were analysed and the results confirm that queue size is only important when small multistage networks are used, where a great traffic exists.

6. CONCLUSIONS.

In this paper several networks used to interconnect multiprocessor systems are studied by means of simulation techniques. This networks are multistage and operate in a packet communication environment. A simulator has been developed under a parametric design to analyse this kind of networks. It is observed that this networks produce considerable improvement in effective bandwidth compared to that of circuit switching multistage networks, surpassing in some cases that of Crossbar network. Bandwidth increases with the number of cycles a request stays in memory, except when saturation is produced in the network. It is shown that the use of smaller multistage networks by connecting

several processors and memory modules to each of the input and output lines of the multistage network produces very interesting results, specially when memory modules are buffered. If we use a second network to eliminate conflicts among requests going to and returning from memory modules a considerable increase in bandwidth is obtained, specially for low values of memory service time.

7. REFERENCES

[1] **D.M. Dias**, and **J.R. Jump**, "Analysis and Simulation of Buffered Delta Networks", IEEE Transactions on Computers, Vol. C–30, Num. 4, April 1981, pp. 273-282.

[2] **G.S. Fishman**, "Principles of Discrete Event Simulation", John Wiley and Sons, 1978.

[3] **A. Gottlieb et al.**, "The NYU Ultracomputer – Designing an MIMD Shared memory Parallel Computer", IEEE Transactions on Computers, Vol. C-32, Num. 2, Feb. 1983, pp. 175-189.

[4] **D.L. Iglehart**, "The Regenerative Method for Simulation Analysis", Current Trends in Programming Methodology, Vol. 3: Software Modelling, Ed. K.M. Chandy, and R.T. Yeh, Prentice Hall, 1978.

[5] **M. Kumar, D.M. Dias** and **J.R. Jump**, "Switching Strategies in a Class of Packet Switching Networks", Proc. 10th Annual Int. Symp. on Computer Architecture, Stockholm, Sweden, 1983, pp. 284-300.

[6] **T. Lang**, "Interconnections Between Processors and Memory Modules Using the Shuffle–Exchange Network", IEEE Trans. on Computers, Vol. C-25, Num. 5, May 1976, pp. 496-503.

[7] **D. Lawrie**, "Access and Alignment of Data in an Array Processor", IEEE Trans. on Computers, Vol. C-24, Num. 12, December 1975, pp. 1145-1155.

[8] **V. López de Buen**, "SIRI. A Multistage Interconnection Networks Simulator", International Symposium on Mini and Microcomputers and their applications, Sant Feliu de Guixols, Gerona, Spain, June 1985, pp. 467-472.

[9] **V. López de Buen**, "Simulation of Multistage Interconnection Networks in a Packet Communication Environment", Int. Conference on Applied Simulation and Modelling, Vancouver, Canada, June 1986, pp. 528-532.

[10] **V. López de Buen**, "Análisis y Simulación de Redes de Interconección en Sistemas Multiprocesador", Tesis Doctoral, Facultat d'Informática,

Universitat Politècnica de Catalunya, Barcelona, Spain, 1986.

[11] **T.E. Moeller** and **P.D. Welch**, "A Spectral Based Technique for Generating Confidence Intervals from Simulation Outputs", Research Report, IBM Thomas J.Watson Research Center, Yorktown Heigths, July 1977.

[12] **D.S. Parker, Jr.**, "Notes on Shuffle/Exchange -Type Switching Networks", IEEE Transactions on Computers, Vol. C-29, Num. 3, March 1980, pp. 213-222.

[13] **J.H. Patel**, "Perfomance of Processor–Memory Interconnections for Multiprocessors", IEEE Trans. on Computers, Vol. C-30, Num. 10, October 1981, pp. 771-780.

[14] **M.C. Pease**, "The Indirect Binary n–Cube Microprocessor Array", IEEE Transactions on Computers, Vol. C-26, Num. 5, May 1977, pp. 458-473.

[15] **H. Siegel**, and **S. Smith**, "Study of Multistage SIMD Interconnection Networks", Proc. 5th Annual Symposium on Computer Architecture, New York, April 1978, pp. 223-229.

[16] **H.J. Siegel**, "Interconnection Networks for SIMD Machines", IEEE, Computer, Vol. 12, Num. 6, June 1979, pp. 57-65.

[17] **B.J. Smith**, "Architecture and Aplications of the HEP Multiprocessor Computer System", Real–Time Signal Processing IV, Vol. 298, Society of Photo–Optical Instrumentation Engineers, August 1981, pp. 241-248.

[18] **B.W. Wah**, "A Comparative Study of Distributed Resource Sharing on Multiprocessors", IEEE Trans. on Computers, Vol. C-33, No. 8, August 1984, pp. 700-711.

[19] **C. Wu**, and **T.Y. Feng**, "On a class of multistage interconnection networks", IEEE Transactions on Computers, Vol. C-29, August 1980, pp. 694-702.