3-1992

PPMB: A Partial-Multiple-Bus Multiprocessor Architecture with Improved Cost-Effectiveness

Hong Jiang
University of Nebraska - Lincoln, jiang@cse.unl.edu

Kenneth C. Smith
Texas A & M University - College Station

PPMB: A Partial-Multiple-Bus Multiprocessor Architecture with Improved Cost-Effectiveness

Hong Jiang and Kenneth C. Smith

Abstract—**This paper addresses the design and performance analysis of partial-multiple-bus interconnection networks. They are bus architectures that have evolved from multiple-bus structure by dividing buses into groups and reducing bus connections. Their effect is to reduce cost and alleviate arbitration and drive requirements without degrading performance significantly. One such structure, called processor-oriented partial-multiple-bus (or PPMB), is proposed. It serves as an alternative to the conventional structure called memory-oriented partial-multiple-bus (or MPMB) and is aimed at higher system performance at less or equal system cost. It has been shown, both analytically and by simulation, that a substantial increase in system bandwidth (up to 20%) is achieved by the PPMB structure over the MPMB structure. With very large systems, the results also imply a significantly improved cost-effectiveness over the conventional multiple-bus architecture.**

Index Terms—**Cost-effectiveness, interconnection network, load-balancing arbitration, multiprocessor architecture, partial multiple-bus structures, performance evaluation.**

I. Introduction

Due to their reliability and cost-effectiveness, multiple-bus structures have assumed considerable importance in both research on, and applications of, interconnection networks in the multiprocessor arena. As a result, a great deal of work has been done in the performance analysis of multiple-bus systems. Such analysis shows that among the three major categories of interconnection networks (i.e., crossbar networks, multistage networks, and multiple-bus networks), multiple-bus structures are the most reliable and, under certain circumstances, the most cost effective [1]–[3], [5], [6], [8]. Nevertheless, multiple-bus structures might still be too costly for very large systems, due to the arbitration and drive requirements they entail.

Lang *et al.* [6] proposed, based on the conventional multiple-bus structure, a new network structure called a partial multiple-bus. The motivation for proposing the new structure was to reduce the cost of the system while trading off an acceptable and tolerable degree of performance degradation. This structure is derived from a conventional multiple-bus structure by dividing memory modules and buses into identical parts (or groups) while maintaining the connection of each processor to every bus. This partial-multiple-bus structure is shown in Fig. 1. As shown in [6], the performance degradation of a partial-multiple-bus is not significant. For a two-group partial-multiple-bus system of size 16 (i.e., $N = M = 16$, where N is the number of processors and M the number of memory modules), the decrease in performance (system bandwidth) is below 6%. For the sake of simplicity and consistency, we shall call this structure memory-oriented partial-multiple-bus, or MPMB.

A different partial multiple-bus structure is proposed in this paper as an alternative to the one proposed by Lang, and as one which provides higher system bandwidth and faster arbitration at lower or equal cost. Derived also from the conventional multiple-bus structure, this structure, called processor-oriented partial multiple-bus, or PPMB, divides processors and buses into identical groups while maintaining the connection of each memory module to every bus.

A notable difference between this structure and the one by Lang is that in it, a memory module has a maximum of B potential paths (where B is the number of buses) to processors while, in Lang's, a memory module has a maximum of only B/g potential paths to processors (where g is the number of groups of buses). This structural difference gives rise to a distinguishing feature of the PPMB structure, namely of having potential for load-balancing arbitration. Load balancing, aimed at fully exploiting the potential for higher bandwidth inherent in the structure, is able to provide a substantial improvement in system performance. As a matter of fact, analytical and simulation results have both shown a maximum of 20% increase in system bandwidth of the PPMB over MPMB. Meanwhile, the cost of a PPMB system has been shown in general to be less than or equal to that of an MPMB of the same size. Note that while the partial-multiple-bus structure, proposed by Lang, was motivated to reduce cost and arbitration time without reducing system bandwidth significantly, we have shown as well that the PPMB structure can lead to a substantial improvement in cost-effectiveness when system size is very large.

In the section that follows, details of the PPMB structure and its load-balancing feature are discussed on a comprehensive basis. Section III introduces probabilistic models for evaluating synchronous-system bandwidth of the structures under study and comparisons are made between PPMB and MPMB. The numerical results produced by them all lie within $\pm 3\%$ of the results of simulation, implying a high level of confidence in the models. Finally, some concluding remarks are given in Section IV.

II. Processor-Oriented Partial Multiple-Bus Structure (PPMB)

A. The Structure

In PPMB, shown in Fig. 2, N processors are divided into g groups with each group of (N/g) processors *fully* connected to a set of (B/g) buses, whereas all M memory modules are connected to all B buses. This is to be contrasted with MPMB in which the M memory modules are divided into g groups where each group of (M/g) memory modules is *fully* connected to a set of (B/g) buses, and all of the N processors are *fully* connected to all buses. For both MPMB and PPMB, g is assumed to be a factor of both B and M (or N).

In the rest of this paper on the study, we will refer to an $N \times M \times B/g$ system as a partial multiple-bus system that has B buses, M memory modules, N processors, and is divided into g groups. In addition, we will replace the notation M/g, N/g, and B/g with MG, NG, and BG, respectively.

Manuscript received May 10, 1989; revised October 23, 1990.

H. Jiang was with the Department of Computer Science, Texas A&M University, College Station, TX 77843. He is now with the Department of CS&E, University of Nebraska, Lincoln, NE 68588.

K. C. Smith is with the Department of Electrical Engineering and Computer Science, University of Toronto, Toronto, Ont., M5S 1A4 Canada.

IEEE Log Number 9102592.

Digital Object Identifier: 10.1109/12.127450

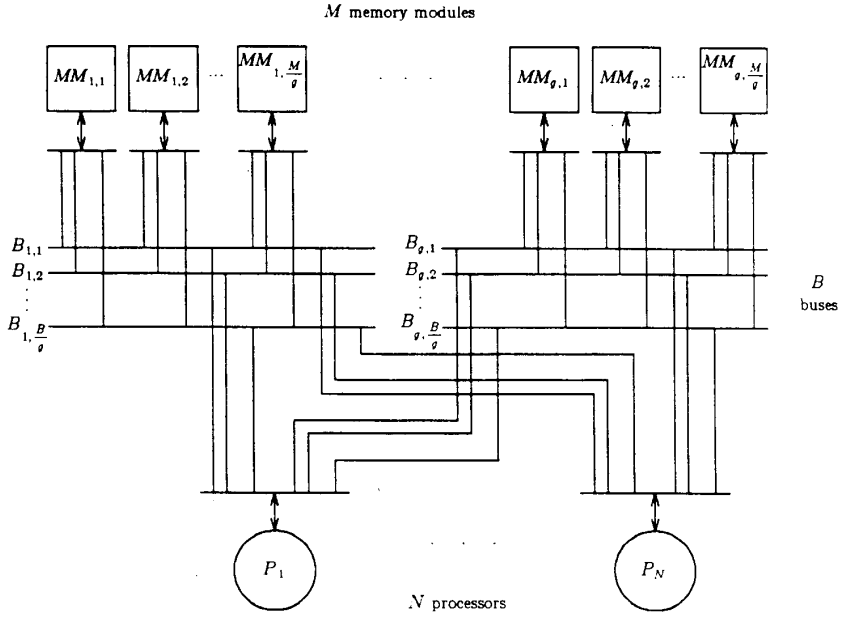

Fig. 1. MPMB structure.

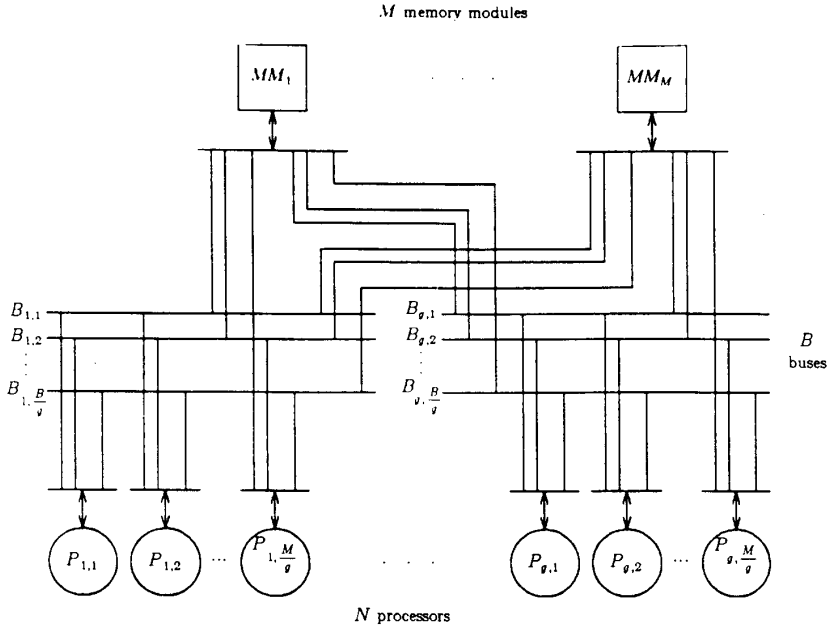

Fig. 2. PPMB structure.

One of the important issues in designing a multiple-bus is how to control the traffic flow in the network. A mechanism for handling traffic control is often referred to as an arbitration scheme.

B. Load-Balancing Arbitration

As a widely accepted arbitration mechanism, a two-level arbitration scheme, proposed by Lang *et al.* [7], is assumed for the MPMB structure in this study. The scheme operates in a $N \times M \times B/g$ system as follows. Associated with each memory module is an N-*user* $\rightarrow$ 1-*server* type arbiter, since there are N demand inputs (each from a single processor) and only one can be granted. This arbiter performs the first level of arbitration that selects one among the processors that require a particular memory. Once this is done, g MG-*user* $\rightarrow$ BG-*server* type arbiters, one for each pair of groups of memory modules and buses, then carry out the second level of arbitration that selects, within each pair of groups, $\min(\mathrm{BG}\ J)$ of the

J memory modules that have at least one outstanding request. Therefore, for a $N \times M \times B/g$ system, the arbitration mechanism is composed of basically M *N-user $\rightarrow$ 1-server* type arbiters and g *MG-user $\rightarrow$ BG-server* type arbiters. Different designs of *N-user $\rightarrow$ 1-server* type arbiters and *M-user $\rightarrow$ B-server* type arbiters can be found in the literature [7].

It is observed, however, that in a PPMB system, a memory module with outstanding requests may be granted a bus in any of the g groups, depending on the processor from which the accepted request is made. This is true simply because any memory module is connected to all groups of buses. In contrast, in an MPMB system, a memory module with an outstanding request can only be granted a bus from the group to which the memory module belongs.

This distinguishing feature of PPMB makes the arbitration scheme employed in MPMB no longer suitable, giving rise to the need for a new one. This feature also suggests that the new scheme should be load-balancing, such that the memory module that has outstanding requests should always be granted a bus, as long as there is at least one free bus in a group to which any of the memory's requesting processors belongs. This is possible since when a memory fails to win the arbitration in one group, it can (literally) always participate in the arbitration process of other groups where its other requesting processors (if any) belong. In other words, memory requests are accepted in such a way that the processors generating the accepted requests are distributed in the most balanced way possible among different groups. The new scheme is thus called *load-balancing* arbitration. Due to the lack of the space in this paper, details of the design and implementation of the load-balancing arbiter, given in [5], will not be presented here. However, an outline of them is sketched, in order to provide the reader with a better insight into the proposed structure.

These are two levels of arbitration. The first level selects one request from each memory queue (if nonempty) as a participant for the second level of arbitration. Each memory module is associated with an arbiter, called a First-Level-Arbiter (FLA_i). An FLA consists of g *NG-user $\rightarrow$ 1-server* type arbiters ($NG1A_j$) and a logic component LB performing the load-balancing function. The FLA takes N inputs, one from each processor, as request lines and another g sets of inputs, $\log_2 MG$ in number, for use by the LB logic. Each $NG1A_j$ performs arbitration among the competing processors of its corresponding group. Outputs of all $NG1A_j$'s are then used as inputs to the LB logic. The LB logic decides, based on the "Least Demanded Group First" (LDGF) policy, which one of the first round winners [outputs of $NG1A_j$'s, i.e., memory requests from different processor groups (if any)] is to participate in the second-level arbitration, and outputs the group number gn_i which designates where the final winner (if any) belongs. If there is such a winner, FLA_i raises a binary signal D_i, indicating that memory M_i is demanding a bus from group gn_i. The LDGF policy simply says that among the first-round winners, the one whose processor group has requested the least number of memory modules in the current bus cycle is selected as the final winner of the first-level arbitration.

The Second-Level-Arbiter (SLA) is composed of M combinational modules $MB(i)$ that perform the assignment of the $g \times BG$ buses, and a state-register which stores the state of the arbiter after each assignment subcycle. It takes the outputs of the FLA_i's as its inputs. The M MB modules, interconnected in a ring fashion by lines carrying arbitration information in combination with bypassing switches, function at any given arbitration cycle as k ($k \leq g$) embedded *NG-user $\rightarrow$ BG-server* type arbiters that are dynamically distributed amoung the M MB modules. Each such *NG-user $\rightarrow$ BG-server* arbiter is associated with, and arbitrates on, a group that has more than BG memory modules demanding its buses. The outputs of SLA give the locations of the granted buses and the corresponding memory modules to which they are assigned.

The load-balancing arbitration mechanism, together with the structure of PPMB, is shown to improve the system performance substantially.

It has been shown that the cost of a multiple-bus can be approximately estimated in terms of bus connections [5], [6]. For instance, the cost of an $N \times M \times B$ system can be said to be proportional to $B(M + N)$. This measure can be directly adapted to partial multiple-bus systems MPMB and PPMB, of size $N \times M \times B/g$, with resulting costs being $B(N + (M/g))$ for MPMB and $B(M + (N/g))$ for PPMB.

It is apparent that one design can be more costly than the other depending on the values of N and M. However, simulations performed in this study and in the literature [6], [8] have shown that, with B fixed, the increase in M beyond N, results in very insignificant improvement in system performance. As well, these simulations indicate that the effect on performance due to a change in M within the range $[N/2, N]$ is much lower than that in the range $[0, N/2]$. Therefore, in applications it is wise to choose M and N such that $N/2 \leq M \leq N$. This implies that $(M + (N/g)) \leq (N + (M/g))$, indicating that the cost of PPMB can be generally less than, or equal to, that of MPMB.

III. Performance Analysis

Performance measures of system bandwidth will now be described. Here, system bandwidth is defined as the expected number of busy buses in each bus cycle. Mathematical models are introduced for these performance measures of the PPMB system. For the purpose of comparison, a probabilistic model developed by Das and Bhuyan [4] is employed to produce numerical results for the MPMB system.

Assumptions: The general assumptions incorporated in the analysis are the following:

1) The processors are synchronized;
2) The memory requests are independent and uniformly distributed random variables;
3) The cycle time of all the memory modules is the same and constant;
4) A processor issues a new request in the next cycle with probability p, after receiving memory service. Probability p is also the request rate, taking the bus cycle time as the basic unit;
5) The propagation delays and arbitration times associated with the interconnection network are not included explicitly but may be thought of as forming part of the memory cycle;
6) Buses are assumed to be assigned at random to the memory modules that have at least one outstanding request. This is done on a cyclic basis;
7) For each memory module that has been granted a bus, a processor is selected at random (also on a cyclic basis) from those with outstanding requests for that module. Other processors are blocked and may request again during the next cycle.

Probabilistic Model: Here we further assume that the requests issued in any cycle are independent of those of the previous cycle. This implies that a rejected request is discarded, rather than being resubmitted in the next cycle.

Now consider a $N \times M \times B/g$ system, regardless of orientation, with p defined as above. The probability of processor P_i requesting memory module M_j, for $1 \leq i \leq N$ and $1 \leq j \leq M$, is given by p/M. It follows that the probability of P_i not requesting M_j is $(1 - (p/M))$. Furthermore, the probability of none of the N processors requesting M_j is given by $(1 - (p/M))^N$. Therefore, the probability that M_j is requested by at least one processor is

given by

$$q = 1 - \left(1 - \frac{p}{M}\right)^N. \tag{1}$$

Supposing that i memory modules have outstanding requests, it is necessary to consider all possible ways of distributing i memory modules among g groups, since it is equally likely that any of the i memory modules may have been requested by any one of the N processors. In addition, the fact that the arbitration is load-balancing-oriented must also be taken into account.

Now let us first find the expression for the number of ways that i items are distributed among g groups of NG places, given that each place can only hold one item. The expression is derived in a constructive way:

$$N(i) = \sum_{G_1=0}^{\min(NG,i)} \binom{NG}{G_1} \sum_{G_2=0}^{\min(NG,i-G_1)} \binom{NG}{G_1} \cdots \cdots \sum_{G_{g-1}=0}^{\min(NG,i-G_1-\cdots-G_{g-2})} \binom{NG}{G_{g-1}} \cdot f(G_g) \tag{2}$$

where

$$f(G_g) = \begin{cases} \binom{NG}{G_g} & 0 \le G_g \le NG \\ 0 & \text{otherwise.} \end{cases}$$

For each combination $\overline{G} = (G_1, \cdots, G_g)$, G_1 is the number of memory modules (out of i) that have been requested by processors from *group*$_1$, and $\binom{NG}{G_1}$ is the number of ways of selecting G_1 processors from NG; G_2 is the number of memory modules (also out of i) that have been requested by processors from *group*$_2$, and $\binom{NG}{G_2}$ is the number of ways of selecting G_2 processors from NG; and so on, and so forth. Therefore, the number of buses that will be assigned to the i memory modules, given a combination $\overline{G} = (G_1, \cdots, G_g)$, is given by

$$bus(\overline{G}, i) = \sum_{l=1}^{g} \min(BG, G_l) \tag{3}$$

with probability

$$p(\overline{G}, i) = \prod_{l=1}^{g} \frac{\binom{NG}{G_l}}{\binom{N}{i}}.$$

Given that there are exactly i memory modules being requested, the mean number of buses that will be assigned with memory modules, is therefore

$$bus(i) = \sum_{all\ \overline{G}} bus(\overline{G}, i) = \sum_{all\ \overline{G}} \frac{\prod_{l=1}^{g} \binom{NG}{G_l}}{\binom{N}{i}} \cdot \sum_{l=1}^{g} \min(BG, G_l). \tag{4}$$

Since $N(i)$, as given in (2), produces all possible combinations $\overline{G}$, thus (4) can be explicitly expressed as

$$bus(i) = \frac{1}{\binom{N}{i}} \left[\sum_{G_1=0}^{\min(NG,i)} \binom{NG}{G_1} \cdots \sum_{G_{g-1}=0}^{\min(NG,i-G_1-\cdots-G_{g-2})} \cdot \left[\binom{NG}{G_{g-1}} f(G_g) \sum_{l=1}^{g} \min(BG, G_l) \right] \right].$$

Note that this does not mean that (2) as a whole is multiplied by the sum $\sum_{l=1}^{g} \min(BG, G_l)$. Instead, it means that each sum of a particular combination is multiplied by one of (2)'s product terms for the same combination.

To take into account the load-balancing effect, (3) is replaced by the following expression

$$bus(\overline{G}, i) = \sum_{l=1}^{g} \min(BG, G_l) + \sum_{j=1}^{Z} \min(Y, j) \binom{Z}{j} q_1^j (1 - q_1)^{Z-j} \tag{5}$$

where

$$Z = \sum_{k=1}^{g} \phi(G_k)(G_k - BG)$$

and

$$\phi(G_k) = \begin{cases} 1 & G_k \ge BG \\ 0 & \text{otherwise} \end{cases}$$

$$Y = \sum_{k=1}^{g} (1 - \phi(G_k))(BG - G_k)$$

$$q_1 = 1 - \left(1 - \frac{p}{M}\right)^{gn}$$

and

$$gn = N - NG \sum_{l=1}^{g} \phi(G_l) - \sum_{l=1}^{g} (1 - \phi(G_l)) \cdot G_l.$$

Z is the number of memory modules, in the combination $\overline{G}$, that would not be assigned with buses if the load-balancing mechanism were not employed. Y is the number of buses still available. For the convenience of later discussion, let $\overline{Z}$ be a set containing exclusively those Z modules, and $\overline{Y}$ be a set containing exclusively those Y buses. q_1 is the probability that any one of the Z memory modules described above is requested at the same time by at least one processor from a group that still has free buses, under the given conditions.

According to the load-balancing policy, a memory module in $\overline{Z}$ may be granted a bus in $\overline{Y}$ as long as it is requested by a processor from a group to which buses in $\overline{Y}$ belong. If the number of such memory modules is less than or equal to Y, all of them are granted buses. Otherwise, only Y of them can be assigned with buses. The second such term of the right-hand side of (5) gives the expected number of such memory modules being granted buses.

Finally, the bandwidth of PPMB is given by the following expression:

$$BW_{\text{PPMB}} = \sum_{i=1}^{M} bus(i) \binom{M}{i} q^i (1 - q)^{M-i} \tag{6}$$

where q is given in (1).

A bandwidth expression for the MPMB system is given as [4]:

$$BW_{\text{MPMB}} = g \cdot MG \cdot q - g \cdot \sum_{i=BG+1}^{MG} (1 - BG) p(i). \tag{7}$$

Improved Model: Because of the assumption that any rejected request is discarded, the models in the previous section tend to underestimate the system bandwidth. If a rejected request is resubmitted in the next cycle, then (intuitively) the rate at which a processor issues requests is higher than it would be otherwise.

To take this fact into account, and thus make the analytical model more realistic and accurate, Yen *et al.* [9] proposed a method called the Steady-State Flow Approach to model the memory interference in synchronous multiprocessor systems. We now adapt it to the partial multiple-bus case to modify our analytical models. The basic idea

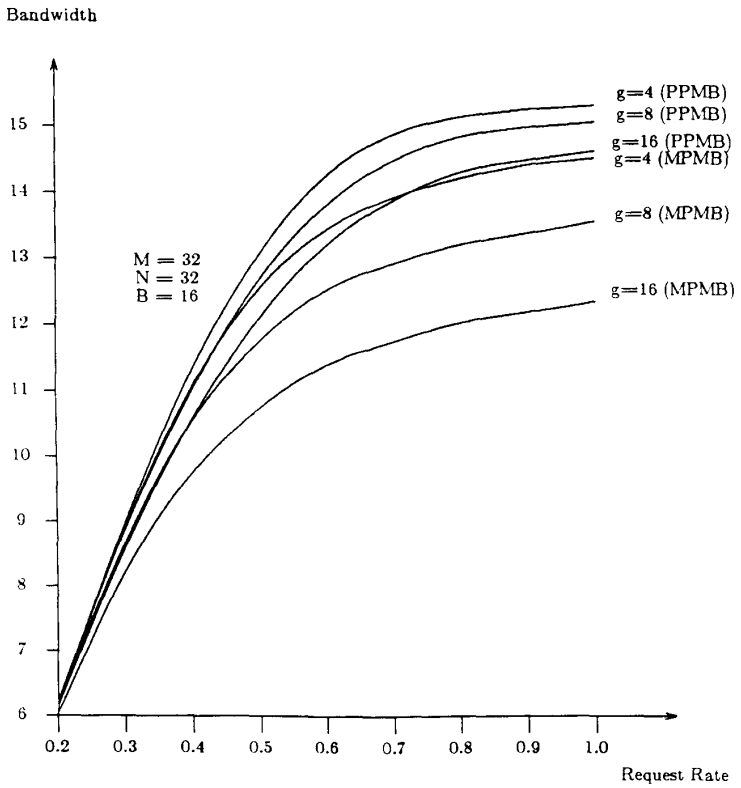

Fig. 3. Bandwidth as a function of request rate.

is to modify q, the probability that there is at least one request for a particular memory module as defined earlier, so as to reflect the effect of the blocked requests that are to be resubmitted. A modified q is given as follows:

$$q = 1 - \left(1 - \frac{fp}{M}\right)^N \left(1 - \frac{1 - \left(1 - \frac{1-f}{M}\right)^N}{M}\right)^M \tag{8}$$

where f is a degradation factor for system performance and also the processor utilization in the steady state. The first product term on the right side of (8) represents the probability that none of the processors has a request for a particular memory module, whereas the second product term is an estimate of the probability that there is no blocked (queued) request for a particular memory module.

Finally, the analytical models in the previous section are modified by replacing the expression for q in (6) and (7) for the PPMB and MPMB systems, respectively, by (8), and then the equation

$$BW(f) = f \cdot N \cdot p$$

is solved for f by iteration using Newton's method. Here $BW(f)$ is the bandwidth expression, and f is initially set to one.

Numerical Results: Numerical results produced by the improved model are displayed in Table I for the PPMB system, and are compared with the results of simulation. As shown, agreement is very good. In fact, all results shown are within $\pm 3\%$ of the results of simulations, a significant improvement over the unimproved models.[1]

[1] Analytic results of this study for the PPMB and [8] and [2] the MPMB, using the unimproved models, indicate errors within 7% of the simulation results.

TABLE I
BANDWIDTH OF PPMB SYSTEM

$N = M = 32$, $B = 16$, $g = 4$

p	Analytical Results	Simulation Results	Percentage Error
0.1	3.1833	3.14	+1.37
0.2	6.2519	6.233	+0.3
0.3	9.0365	9.048	−0.12
0.4	11.318	11.422	−0.91
0.5	12.943	13.1692	−1.7
0.6	13.943	14.354	−2.7
0.7	14.55	14.979	−2.8
0.8	14.911	15.238	−2.1
0.9	15.132	15.354	−1.4
1.0	15.277	15.4264	−0.9

PPMB System Versus MPMB System: Based on simulation results, Table II shows the degree of performance improvement of the PPMB structure over the MPMB structure.

A maximal increase of almost 20% in system bandwidth is achieved (see Table II) while cost remains the same and could even be decreased in applications where $M < N$, as discussed in Section II. Further, Fig. 3 shows another feature that further evidences the cost-effectiveness of the PPMB structure. As we can see in the figure, a $32 \times 32 \times 16/4$ MPMB system is equivalent (or even a little inferior) to a $32 \times 32 \times 16/16$ PPMB system. However, the cost of such a PPMB system is a lot lower than that of the MPMB system. Recall that the cost of partial multiple-bus system is in part inversely proportional to the number of groups into which it is divided.

TABLE II
BANDWIDTH IMPROVEMENT OF PPMB OVER MPMB

$N = M = 32, B = 16$				
number of groups	request rates	BW of PPMB	BW of MPMB	percentage improvement
8	0.2	6.2646	6.2300	+1.40
8	0.4	11.0700	10.6700	+3.40
8	0.6	13.8936	12.6168	+10.12
8	0.8	14.9458	13.3140	+12.25
8	1.0	15.1842	13.6804	+11.00
16	0.2	6.1628	6.0486	+1.80
16	0.4	10.6596	9.8634	+8.07
16	0.6	13.2930	11.4652	+16.00
16	0.8	14.4126	12.0196	+19.50
16	1.0	14.7456	12.4504	+18.40

At this point, it is necessary to emphasize that the introduction of the load-balancing arbitration mechanism into PPMB does not necessarily imply an increase in cost nor a decrease in arbitration speed. First of all, the extra logic in the arbiter may indeed increase the complexity, but with present-day VLSI technology, this is unlikely to lead to difficulty with implementation since, as partly shown in Section II and expanded in [5], the total number of wires going into and out of the arbiter is not changed from that conventionally required [7]. Furthermore, the arbitration time may even decrease, because in the new scheme g parallel N/g-*user* $\rightarrow$ *one-server* type arbiters, instead of a single N-*user* $\rightarrow$ *one-server* type arbiter, are used at each memory module for first-level arbitration, an arrangement which likely decreases the arbitration time at that level by almost a factor of g. Moreover, the second-level arbiter of PPMB is virtually a dynamic combination of up to g N/g-*user* $\rightarrow$ M/g-*server* type arbiters, and therefore will not increase arbitration time at this level either.

IV. CONCLUSIONS

The processor-oriented partial-multiple-bus structure (PPMB), proposed here as an alternative to the memory-oriented partial-multiple-bus structure (MPMB), has been shown to improve system performance substantially. It can provide an increase in system bandwidth of up to 20%, without the tradeoff in cost usually demonstrated by alternative systems. That this occurs is not totally surprising in view of the structural difference between these two partial-multiple-bus systems. That is, in a PPMB structure, a memory module has a maximum of B potential paths (where B is the total number of buses) to processors while, in a MPMB structure, a memory module has a maximum of only B/g potential paths (where g is the total number of groups of the partial-multiple-bus) to processors. This potential for improvement of system bandwidth is fully fulfilled by the load-balancing arbitration mechanism, whose positive effect is demonstrated by both analytical results and simulations. While the MPMB structure was motivated to reduce the cost of a very large system without degrading its performance significantly, the PPMB structure will evidently substantiate this perspective by outperforming MPMB itself. To further increase the cost-effectiveness of the structures under study, however, future research must be directed to incorporating a cache mechanism into the structures (both PPMB and MPMB) and analyzing its effect on system performance.

REFERENCES

[1] D. P. Bhandarkar, "Analysis of memory interference in multiprocessors," *IEEE Trans. Comput.*, vol. C-24, pp. 897–908, Sept. 1975.
[2] L. N. Bhuyan, "A combinatorial analysis of multibus multiprocessors," in *Proc. '84 Int. Conf. Parallel Processing*, pp. 225–232.
[3] ——, "An analysis of processor-memory interconnection networks," *IEEE Trans. Comput.*, vol. C-34, pp. 279–283, Mar. 1985.
[4] C. R. Das and L. N. Bhuyan, "Bandwidth availability of multiple-bus multiprocessors," *IEEE Trans. Comput.*, vol. C-34, pp. 918–926, Oct. 1985.
[5] H. Jiang, "Partial-multiple-bus computer structures with improved cost-effectiveness," M.A.Sc. Thesis, Dep. Elec. Eng., Univ. of Toronto, Jan. 1987.
[6] T. Lang *et al.*, "Bandwidth of crossbar and multiple-bus connections for multiprocessors," *IEEE Trans. Comput.*, vol. C-31, pp. 1227–1233, Dec. 1982.
[7] T. Lang and M. Valero, "M-users B-servers arbiter for multiple-buses multiprocessors," *Microprocessing and Microprogramming*, North-Holland Publishing Company, 10, 1982, pp. 11–18.
[8] Q. Yang, "Communication performance in multiple-bus systems," M.A.Sc. Thesis, Dep. Elec. Eng., Univ. of Toronto, 1985.
[9] D. W. L. Yen, J. H. Patel, and E. S. Davidson, "Memory interference in synchronous multiprocessor systems," *IEEE Trans. Comput.*, vol. C-31, pp. 1116–1121, Nov. 1982.