A PARTIAL-MULTIPLE-BUS COMPUTER STRUCTURE WITH IMPROVED COST-EFFECTIVENESS

Hong Jiang
The Center for Advanced Computer Studies
University of Southwestern Louisiana
P.O.Box 44330
Lafayette, LA 70504-4330

Kenneth C. Smith
Depts. of Ele. Eng. and Computer Science
University of Toronto
Toronto, Ont.
M5S 1A4 CANADA

ABSTRACT

This paper addresses the design and performance analysis of partial-multiple-bus interconnection networks. 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. PPMB's structural feature, which distinguishes itself from the conventional, is to provide every memory module with B paths to processors (where B is the total number of buses). This in contrast to the mere $\frac{B}{g}$ paths provided in the conventional MPMB structure (where g is the number of groups), suggests a potential for higher system bandwidth. This potential is fully fulfilled by the load-balancing arbitration mechanism suggested, which in turn highlights the advantages of the proposed structure. As a result, 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. In addition to the fact that the cost of PPMB is less than, or equal to, that of MPMB, its reliability is shown to be slightly increased as well.

1. Introduction.

Due to their reliability and cost-effectiveness, multiple-bus structures have assumed considerable importance in both research on, and applications of, interconnection networks. As a result, a great deal of work has been done in the performance analysis of multiple-bus systems. Such analysis shows that amongst 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. [2, 3, 4, 7, 11, 6, etc.] 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 [7] 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 [7], the performance degradation of a partial-multiple-bus is quite acceptable. 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 is below 6%. For the sake of simplicity and consistency, we shall call this structure Memory-Oriented Partial-Multiple-Bus, or MPMB.

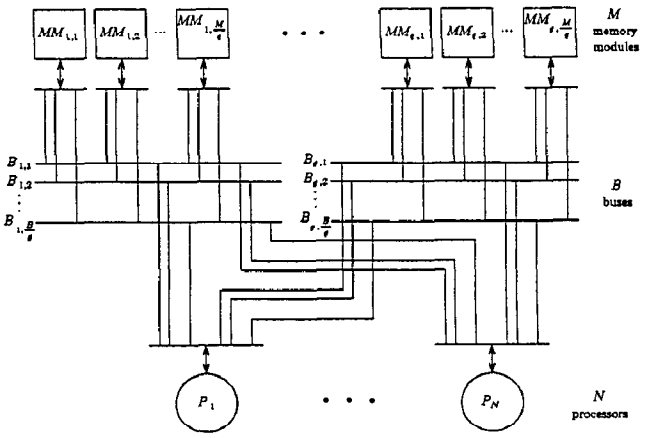

Fig. 1. MPMB Structure

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 improved cost-effectiveness. 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 $\frac{B}{g}$ potential paths to processors (where g is the number of groups). 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. Some improvement in reliability is also achieved as indicated by analytical results. 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.

In the section that follows, details of the PPMB structure and its load-balancing feature are discussed on a comprehensive basis. Section 3 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 ±3% of the results of simulation, implying a high level of competence in the models. Finally, some concluding remarks are given in section 4.

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

The Structure

This structure is so named because of the fact that the processors and buses are divided into corresponding groups, as shown in Fig. 2.

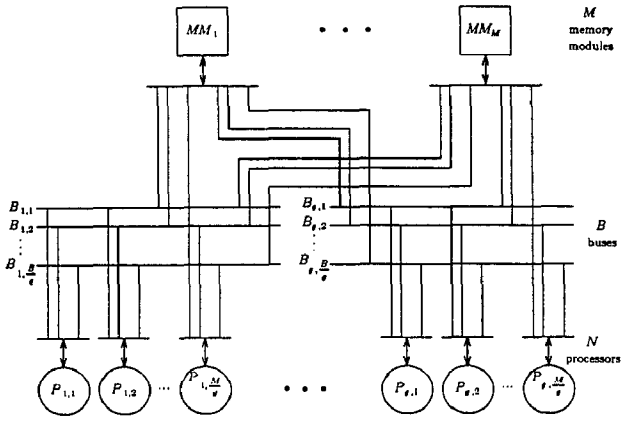

Fig. 2. PPMB Structure

In PPMB, N processors are divided into g groups with each group of $(\frac{N}{g})$ processors *fully* connected to a set of $(\frac{B}{g})$ buses, whereas all M memory modules are connected to all buses. This is to be contrasted with MPMB in which the M memory modules are divided into g groups where each group of $(\frac{M}{g})$ memory modules is *fully* connected to a set of $(\frac{B}{g})$ 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 report 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 $\frac{M}{g}$, $\frac{N}{g}$, and $\frac{B}{g}$ with MG, NG, and BG respectively.

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 on arbitration scheme.

As a widely accepted arbitration mechanism, a two-level arbitration scheme, proposed by Tomes Lang et al [8], 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 $N\text{-}user \rightarrow 1\text{-}server$ 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 $MG\text{-}user \rightarrow BG\text{-}server$ 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, $min(BG, J)$ 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 $N\text{-}user \rightarrow 1\text{-}server$ type arbiters and g $MG\text{-}user \rightarrow BG\text{-}server$ type arbiters. Different designs of $N\text{-}user \rightarrow 1\text{-}server$ type arbiters and $M\text{-}user \rightarrow B\text{-}server$ type arbiters can be found in the literature [8].

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 of 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 amongst 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 [6], 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.

There 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 $NG\text{-}user \rightarrow 1\text{-}server$ type arbiters ($NG\,1A_j$) 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 to store the state of the arbiter after each assignment subcycle. It takes the outputs of the FLA_is as its inputs. The M *MB* modules, interconnected in a ring fashion by lines carrying arbitration information in combination with by-passing switches, function at any given arbitration cycle as k $(k \leq g)$ embedded *NG-user→BG-server* type arbiters that are dynamically distributed among the M *MB* modules. Each such *NG-user→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 their 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.

Estimate of Cost

To estimate the cost of the structure under study, one should take into account factors such as the number of connections, the number of wires, the capacitive load supported by each bus, and the number of arbiters and their complexity. For an $N \times M \times B$ multiple-bus system, in general, the number of connections is proportional to $B(M+N)$, the number of wires is proportional to B, and each bus supports a capacitive load proportional to $M+N+K(M+N-1)$. The value of K is dependent on the technology, and for present-day tristate circuits, is much smaller than one [7]; therefore, the capacitive load is essentially proportional to $(M+N)$. The cost of the arbiters may be considerably small in comparison to other factors (i.e. connections, wires, etc), due to the rapid development of VLSI technology. Therefore, for simplicity, the cost of arbiters will not be considered in the present study.

Combining the above arguments, the cost of an $N \times M \times B$ system can be said to be proportional to $(B(M+N)+B+M+N) = B(M+N+1)+M+N$, approximately $(B+1)(M+N+1)$, or simply $B(M+N)$.

The above result can be directly adapted to partial multiple-bus systems MPMB and PPMB, of si e $N \times M \times B/g$, with the following result:

-- the cost of MPMB is proportional to $B\left(N+\frac{M}{g}\right)$;

-- the cost of PPMB is proportional to $B\left(M+\frac{N}{g}\right)$.

Simulations performed in this study and in the literature [7, 11, etc] 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 $[\frac{N}{2},N]$ is much lower than that in the range $[0,\frac{N}{2}]$. Therefore, in applications it is wise to choose M and N such that $\frac{N}{2} \leq M \leq N$.

For this condition, we have: $\left(M+\frac{N}{g}\right) \leq \left(N+\frac{M}{g}\right)$ indicating the cost of PPMB to be generally less than, or equal to, that of MPMB.

3. 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 measures of both systems for the purpose of comparison.

Assumptions

The general assumptions incorporated in the analysis are the following:

a). The processors are synchronized;

b). The memory requests are independent and uniformly distributed random variables;

c). The cycle time of all the memory modules is the same and constant;

d). 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;

e). 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;

f). 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;

g). 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 models

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 $\frac{p}{M}$. It follows that the probability of P_i not requesting M_j is $(1-\frac{p}{M})$. Furthermore, the probability of none of the N processors requesting M_j is given by $\left(1-\frac{p}{M}\right)^N$. Therefore,

the probability that M_j is requested by at least one processor is given by

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

This is the case for both the MPMB and PPMB systems.

Now we are ready to derive expressions for system bandwidth for both the MPMB and PPMB systems:

1). MPMB System :

A probabilistic model developed by Das and Bhuyan [5] is employed here for the MPMB system. The bandwidth is constrained by BG, the number of buses in each group. Since all groups are identical, we need only consider one of them.

Given the probability of a memory module being requested in equation (1), the probability $p(i)$ that exactly i memory modules are requested in a cycle is given by

$$p(i) = \binom{MG}{i} q^i \left(1-q\right)^{MG-i}$$

where $\binom{MG}{i}$ is the number of ways of choosing i memory modules from MG memory modules.

Since a multiple-bus system with BG buses can allow at most BG processor-memory connections per cycle, the system becomes saturated when more than BG requests are generated, and will allow only BG connections simultaneously. Therefore, the bandwidth of the subsystem (group) is given as:

$$\begin{aligned} BW_{grp} &= \sum_{i=1}^{BG} i \cdot p(i) + \sum_{i=BG+1}^{MG} BG \cdot p(i) \\ &= \sum_{i=1}^{MG} i \cdot p(i) - \sum_{i=BG+1}^{MG} \left(i-BG\right) p(i) \\ &= MG \cdot q - \sum_{i=BG+1}^{MG} \left(i-BG\right) p(i) \end{aligned} \tag{2}$$

Multiplying (2) by g, we get a bandwidth expression for the MPMB system:

$$\begin{aligned} BW_{MPMB} &= g \cdot BW_{grp} \\ &= g \cdot MG \cdot q - g \cdot \sum_{i=BG+1}^{MG} \left(i-BG\right) p(i) \end{aligned}$$

2). PPMB System :

It is more complicated to derive an expression for the bandwidth of PPMB: 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:

$$\begin{aligned} N(i) = &\sum_{G_1=0}^{\min(NG,i)} \binom{NG}{G_1} \sum_{G_2=0}^{\min(NG,i-G_1)} \binom{NG}{G_2} \cdots \\ &\cdots \sum_{G_{g-1}=0}^{\min(NG,i-G_1-\cdots-G_{g-2})} \binom{NG}{G_{g-1}} \cdot f(G_g) \end{aligned} \tag{3}$$

where

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

For each combination $\overline{G} = (G_1, \ldots, G_g)$, 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 $\overline{G} = (G_1, \ldots, G_g)$, is given by

$$bus(\overline{G}, i) = \sum_{l=1}^{g} min(BG, G_l) \tag{4}$$

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

$$\begin{aligned} bus(i) &= \sum_{all\ \overline{G}} bus(\overline{G}, i) \\ &= \sum_{all\ \overline{G}} \frac{\prod_{l=1}^{g} \binom{NG}{G_l}}{\binom{N}{i}} \cdot \sum_{l=1}^{g} min(BG, G_l) \end{aligned} \tag{5}$$

Since $N(i)$, as given in (3), produces all possible combinations $\overline{G}$, thus (5) can be explicitly expressed as

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

Note that this does not mean that (3) 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 (3)'s product terms for the same combination.

To take into account the load-balancing effect, equation (4) is replaced by the following expression

$$bus(\overline{G},i) = \sum_{l=1}^{g} min(BG,G_l) + \sum_{j=1}^{Z} min(Y,j)\binom{Z}{j} q_1^j (1-q_1)^{Z-j} \quad (6)$$

where

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

and

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

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

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

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 sum term of the right hand side of (6) gives the expected number of such memory modules being granted buses.

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

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

where q is given in (1).

Improved Models

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 [13] 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 following are the basic ideas employed in this approach.

Suppose there were no interference (i.e. neither memory nor bus conflict occurs). Thus the system bandwidth would be $BW = Np$, where p is the request rate, or the probability of issuing a request in each cycle as defined earlier in the paper. In multiple-bus systems, however, memory and bus conflicts frequently occur. Therefore the bandwidth will be

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

Where f is a degradation factor for system performance and also the processor utilization in the steady state. Thus $f \cdot N$ is the number of processors that are active (engaged in internal computing or accessing cycles), while $N \cdot (1-f)$ is the number of processors that are blocked (in waiting cycles). When the system is in equilibrium, the $f \cdot N$ active processors issue $f \cdot N \cdot p$ new requests each cycle and $f \cdot N \cdot (1-p)$ processors are active in internal computing cycles. Since $f \cdot N \cdot p$ requests are serviced in each cycle, in the steady-state, the number of active (unblocked) processors remains constant at $f \cdot N$.

In order eventually to find q, the probability that there is at least one request for a particular memory module, it is necessary to find r, the number of nonempty memory queues in the steady state. As a first approximation of r, find r', where

$$\frac{r'}{M} = 1-\left(1-\frac{(1-f)}{M}\right)^N$$

This is a result of distributing the $N \cdot (1-f)$ queued requests uniformly among M queues. Here, $\frac{1-f}{M}$ is the probability of a processor being in a particular queue according to a uniform distribution.

Note that r' overestimates r, since multiple rejected requests to the same memory module tend to cause fewer, but longer, nonempty queues than predicted by a uniform redistribution. An improved estimate for r would be obtained by assuming that each of the M queues has probability $\frac{r'}{M}$ of making an access request, in which case,

$$\frac{r}{M} = 1-\left(1-\frac{\frac{r'}{M}}{M}\right)^M \quad (7)$$

is the estimated probability that there is a queued request for a particular memory module.

The probability that there is no queued request for that memory is $1-\frac{r}{M}$. The probability that none of the processors has a request for that particular memory module is

$$\left(1-\frac{f\ p}{M}\right)^N \quad (8)$$

Therefore, combining (7) and (8), we obtain an expression for q:

$$q = 1-\left(1-\frac{f\ p}{M}\right)^N \left(1-\frac{1-\left(1-\frac{1-f}{M}\right)^N}{M}\right)^M \quad (9)$$

Finally, the analytical models in the previous section are modified by replacing the expression for q in models for the PPMB and MPMB systems by (9), 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 models are displayed in Table-1 and Table-2 for MPMB and PPMB systems respectively and are compared with the results of simulation. As shown, numerical results produced by PPMB and MPMB models are all in very good agreement with simulation. In fact, all results shown are within ±3% of the results of simulations, a significant improvement over the unimproved *models* [1].

Table-1

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

Table-2

Bandwidth of MPMB System
N=M=32, B=16, g=4

p	Analytical Results	Simulation Results	Percentage Error
0.1	3.1833	3.165	+0.57
0.2	6.2506	6.1816	+1.1
0.3	9.0213	8.999	+0.24
0.4	11.251	11.165	+0.76
0.5	12.786	12.654	+1.04
0.6	13.722	13.526	+1.45
0.7	14.267	14.019	+1.78
0.8	14.518	14.347	+1.9
0.9	14.964	14.541	+2.1
1.0	14.9638	14.651	+2.1

1) Analytic results of this study and [11 and 3], using the unimproved models, indicate errors within 7% of the simulation results.

The simulation results are produced by two programs written in PASCAL that simulate the synchronous behaviors of PPMB and MPMB structures respectively. These programs are driven by linear congruential random number generators using the *shuffle* technique. The regenerative method, using an approximate sequential stopping rule proposed by Lavenberg [10], is used in the simulation to control the width of the confidence interval, where the regeneration points are chosen to be those system states where all the memory queues are empty. A confidence level of 95% is achieved in the simulations. [6]

Reliability

While the improved cost-effectiveness brought by the new structure is evident, it is also one of our aims that the reliability of PPMB should not suffer, if not significantly increase. An analysis in [6] has shown that the reliability of PPMB is in fact slightly higher than that of MPMB. In this demonstration a combinatorial model devised in [5] is employed. The model defines, among other things, that the system reliability is the probability that the system is capable of performing a task at a given time, given the numbers of processors, memory modules and buses that are necessary for that task.

PPMB System vs MPMB System

Based on simulation results, Table-3 shows the degree of performance improvement of the PPMB structure over the MPMB structure.

Table-3

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	.2	6.2646	6.2300	+1.40
8	.4	11.0700	10.6700	+3.40
8	.6	13.8936	12.6168	+10.12
8	.8	14.9458	13.3140	+12.25
8	1.0	15.1842	13.6804	+11.00
16	.2	6.1628	6.0486	+1.80
16	.4	10.6596	9.8634	+8.07
16	.6	13.2030	11.4652	+16.00
16	.8	14.4126	12.0196	+19.5
16	1.0	14.7456	12.4504	+18.40

A maximal increase of almost 20% in system bandwidth is achieved (see Table-3) while cost remains the same and could even be decreased in applications where $M < N$, as discussed in section 2. Further, Fig. 3 shows another feature that further evidences the cost-effectiveness of the PPMB structure. As we can see in the figure, a $32 \times 32 \times 16 \mid 4$ MPMB system is equivalent (or even a little inferior) to a $32 \times 32 \times 16 \mid 16$ PPMB system. However, the cost of such a PPMB system is a lot lower than that of a 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.

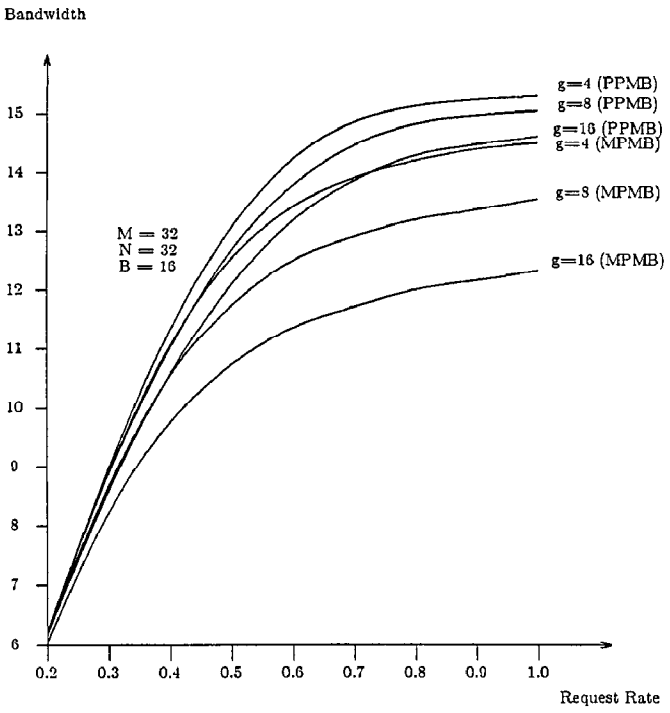

Fig. 3 Bandwidth as a Function of Request Rate

At this point, it is necessary to emphasize the fact, expanded upon in [6], 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.

4. 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 $\frac{B}{g}$ potential paths (where g is the total number of groups of the partial-multiple-bus) to processors. This potential for the improvement of system bandwidth is fully fulfilled by the load-balancing arbitration mechanism, whose positive effect is demonstrated by both analytical results and simulations. As well, the reliability of the PPMB structure is slightly improved over that of MPMB structure. While the significance of this improved cost-effectiveness is obvious, it might, as well, imply the potential for increased applicability of partial-multiple-bus interconnection structures.

ACKNOWLEDGEMENTS

The authors are grateful to Dr. Laxmi N. Bhuyan for his frank and helpful comments on the manuscript. Thanks are extended to the Center for Advanced Computer Studies at USL and to the University of Toronto for their support.

References

[1] Arnold, T. F., "The Concept of Coverage and it's Effect on the Reliability Model of a Repairable System," *IEEE Trans. on Comput.*, C-22, Mar. 1973, pp.251-254.

[2] Bhandarkar, D. P., "Analysis of Memory Interference in Multiprocessors," *IEEE Trans. on Comput.*, Vol. C-24, Sept. 1975. pp 897-908.

[3] Bhuyan, Laxmi N., "A Combinatorial Analysis of Multibus Multiprocessors," *Proc. of 84' Int. Conf. Parallel Processing,* pp 225-232.

[4] Bhuyan, Laxmi N., "An Analysis of Processor-Memory Interconnection Networks," *IEEE Trans. on Comput.*, vol. C-34, no. 3, March 1985, pp 279-283

[5] Das, Chita R. and Bhuyan, Laxmi N., "Bandwidth availibility of Multiple-Bus Multiprocessors," *IEEE Trans. on Comput.* vol. C-34, no. 10, Oct. 1985, pp 918-926.

[6] Jiang, Hong, "Partial-Multiple-Bus Computer Structures with Improved Cost-Effectiveness," M.A.Sc. Thesis, Dept. of Electrical Engineering, University of Toronto. Jan. 1987.

[7] Lang, Tomas, et al, "Bandwidth of Crossbar and Multiple-Bus Connections for Multiprocessors," *IEEE Trans. on Comput.*, Vol. C-31, no. 12, Dec. 1982, pp 1227-1233.

[8] Lang, Tomas and Valero, M., "M-users B-servers arbiter for multiple-buses multiprocessors," *Microprocessing and Microprogramming,* North-Holland Publishing Company, 10, 1982, pp. 11-18.

[9] S. S. Lavenberg and D. R. Slutz, "Introduction to Regenerative Simulation," *IBM J. Res. Develop.* 19, 1975, pp 458-462.

[10] S. S. Lavenberg and C. H. Saucer, "Sequential Stopping Rules for the Regenerative Method of Simulation," *IBM J. Res. Develop.* 21, 1977, pp 545-558.

[11] Yang, Q, "Communication Performance in Multiple-bus Systems", M.A.Sc. Thesis, Dept. of Electrical Engineering, University of Toronto, 1985.

[12] Yang, Q, Zaky, S. G., "Communication Performance in Multiple-Bus Systems," *IEEE Trans. on Comput.*, (to appear).

[13] Yen, D. W. L., Patel, J. H., and Davidson, E. S., "Memory Interference in Synchronous Multiprocessor Systems," *IEEE Trans. on Comput.*, Vol. C-31, No. 11, Nov. 1982, pp. 1116-1121.