

N-way Ring and Square Arbiters

Masashi Imai
The University of Tokyo
miyabi@hal.rcast.u-tokyo.ac.jp

Tomohiro Yoneda
National Institute of Informatics
yoneda@nii.ac.jp

Takashi Nanya
The University of Tokyo
nanya@hal.rcast.u-tokyo.ac.jp

Abstract—In this paper, we propose two new N -way arbiter circuits. One circuit is based on the token-ring arbiters and another circuit is based on the mesh arbiters. The idea of the ring arbiter is to generate a lock signal by a token which is based on the non-return-to-zero signaling. It can achieve low latency and high throughput arbitration for a heavy work load environment. The idea of the mesh arbiter is to perform arbitrations between $N/2$ pairs at the same level and repeat them $N-1$ times. They can issue grant signals fairly. In this paper, we compare the performance of these N -way arbiters using 65nm process technologies qualitatively and quantitatively. We conclude that the proposed mesh arbiters are suitable when the number of inputs is 5 or less. We also conclude that we must select the appropriate arbiters considering trade-off between latency, throughput, area, and energy when the number of inputs is larger than 5.

I. INTRODUCTION

As VLSI technology advances, a large number of resources can be implemented in a chip. Arbitration circuits [1] play an important role to manage shared resources. They allocate a restricted number of resources to different client processes. N -way arbiters which have N request signals and N grant signals issue at most one of N grant signals at any time. The design methods of N -way arbiter circuits have been well studied [1], [2], [3], [4], [5], [6], [7]. Generally, N -way arbiters are composed by two-way mutual exclusion elements (ME) and some standard gates. They are typically based on the standard topologies such as a mesh, cascaded tree or ring [1]. It has been pointed out that mesh arbiters are not suitable for large N while cascaded tree arbiters and ring arbiters are not suitable for small N due to their complexity. However, very few attempts have been made at quantitative comparison between these topologies. Thus, in this paper, we design these arbiters and compare their performance using 65nm process technologies.

In the mesh topologies, there could be a number of possible solutions for N -way arbitration, depending on the connection pattern of two-way MEs. For example, a regular triangle layout has been proposed [1]. However, this triangle mesh arbiter is unfair since it has priority depending on the topological order of input signals. Fairness means that arbiters have the same probability for two simultaneously arriving request signals to issue the grant signals. In this paper, we propose a fair mesh arbiter called “square arbiter” which has no priority. In the ring topologies, a resource is represented by a token in the ring structure. Token-ring based arbiters are typically classified into two categories such as busy ring arbiters and lazy ring arbiters [1]. In busy ring arbiters, a token rotates through the ring constantly, resulting in large energy dissipation. In lazy ring arbiters, a token is

kept in the node where the client was the most recent winner. In this paper, we propose a simple circuit of lazy ring arbiters and evaluate its performance.

The remainder of this paper is organized as follows. We first explain traditional N -way arbiter circuits in Section II. We propose a simple circuit implementation of lazy ring arbiters in Section III and propose a fair mesh arbiter in Section IV. Then, we compare the proposed arbiters with traditional arbiters using 65nm process technologies in Section V. Finally, we describe our conclusions in Section VI.

II. TRADITIONAL N -WAY ARBITERS

A. Cascaded Tree Arbiter

Cascaded tree arbiters are based on a tree topology. Figure 1 shows the overview of N -way tree arbiters. In Fig. 1, boxes which have three input signals $IREQ1, IREQ2, OGR$ and three output signals $IGR1, IGR2, OREQ$ represent the tree arbiter module shown in Fig. 2. A box labeled “ME” represents a two-way mutual exclusion element. In a tree arbiter module, the input signals $IREQ1, IREQ2$ arbitrate and generate the output signal $OREQ$ on their behalf. The output request signals propagate to the next level of the tree and arbitrate with their neighbor in the same fashion as shown in Fig. 1. At the root of the tree whose level is $\lceil \log_2 N \rceil$, an ME is used to resolve arbitration. If N is not 2^n , the tree topology is unfair. Thus, the latency which represents the duration from the time a request signal arises until the time its grant signal arises depends on the level from the root node. The worst latency is proportional to $\lceil \log_2 N \rceil$. Generally, it has been recognized that a two-way mutual exclusion element cannot be constructed using only digital logic gates. In this paper, we use an ME circuit shown in Fig. 1. It may enter a metastable state and stay indefinitely. If some ME circuits are used sequentially, they may spend long time in their metastable states. Thus, the number of cascaded MEs is an important criteria. For N -way cascaded tree arbiters, the maximum number of cascaded MEs is $\lceil \log_2 N \rceil$.

B. Mesh Arbiter

In the mesh topologies, MEs are used for constructing arbitrating combinations on the 2-of- N basis. There could be many possible solutions for N -way arbitration. Figure 3 shows a simple regular layout of N -way mesh arbiter [1]. This layout is obtained from the triangle under the main diagonal of the $N \times N$ matrixes. It avoids mutual crossing of interconnects between MEs. In Fig. 3, number i on the input side of each ME represents successors of primary input REQ_i . The total number of MEs is $N(N-1)/2$. The number of cascaded MEs for each path from a request signal REQ_i

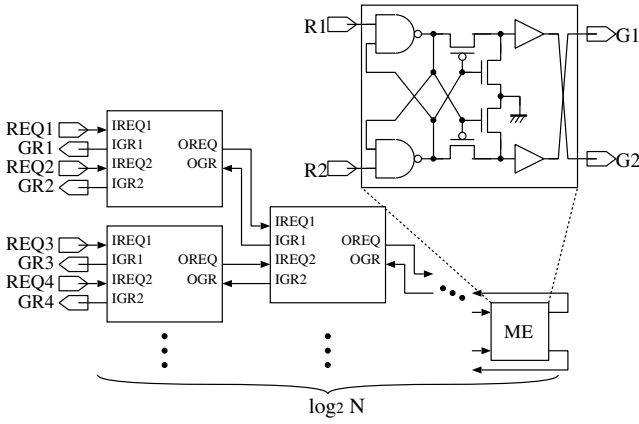


Fig. 1. N-way tree arbiter.

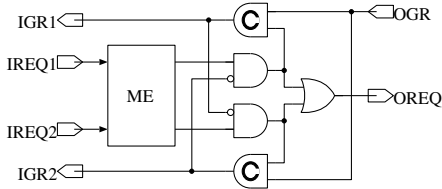


Fig. 2. Tree arbiter module.

to its grant signal GR_i is all $N - 1$. Thus, the latency is proportional to N . This triangle mesh arbiter is unfair since it has priority depending on the topological order of input signals. If all the input signals REQ_1, \dots, REQ_N arise at the same time, either the signal GR_N or the signal GR_{N-1} always arises. Which wins depends on the internal delays of the mesh topologies.

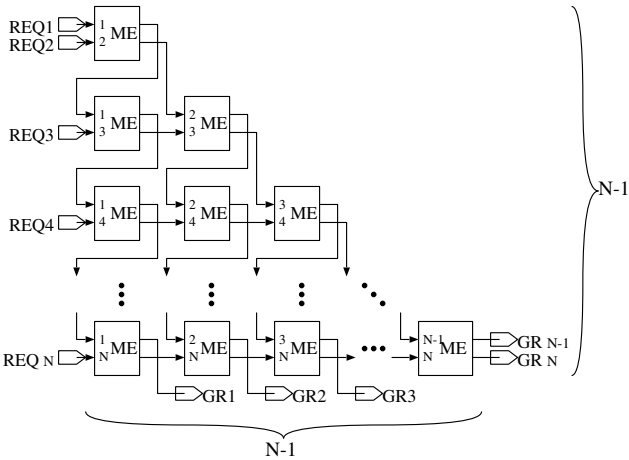


Fig. 3. N-way triangle mesh arbiter.

C. Ring Arbiter

Ring arbiters are based on the principle of a token ring. As mentioned before, token-ring based arbiters are typically classified into two categories such as busy ring arbiters and lazy ring arbiters [1]. Some circuit designs and their comparison have been proposed [1], [4]. The lazy ring

protocol is suitable for a light work load environment. The busy ring arbiter can offer faster response time for a heavy work load environment while it consumes excessive power. Then, some hybrid solutions have been proposed in [4]. In this paper, we propose a simple lazy ring arbiter circuit in the next section.

III. LAZY RING ARBITER WITH NON-RETURN-TO-ZERO TOKEN

In the ring topologies, a token represents one shared resource. In lazy ring arbiters, it is required that the token does not move when all the request signals are inactive. Figure 4 shows the top-level view of the proposed lazy ring arbiter. In Fig. 4, boxes represent the ring arbiter module shown in Fig. 5. In Fig. 4, a NOR gate whose inputs are all the request signals REQ_1, \dots, REQ_N generates a signal NO_REQ which represents whether there are no active requests. As shown in Fig. 5, the signal NO_REQ works as a clock gating signal. When the signal NO_REQ is 1, i.e. there is no active request, the token does not move to the successor module. On the contrary, when the signal NO_REQ is 0, the token can move through the ring. In this implementation, a token which rotates through the ring is based on the Non-Return-to-Zero signaling. It means that the token is represented by both “1” and “0,” physically. As shown in Fig. 5, the signal $lock$ which indicates a logical token becomes 1 when the signal $TOKEN_IN$ is different from the signal $TOKEN_OUT$. In the ring structure, at most one of $lock$ signals becomes 1.

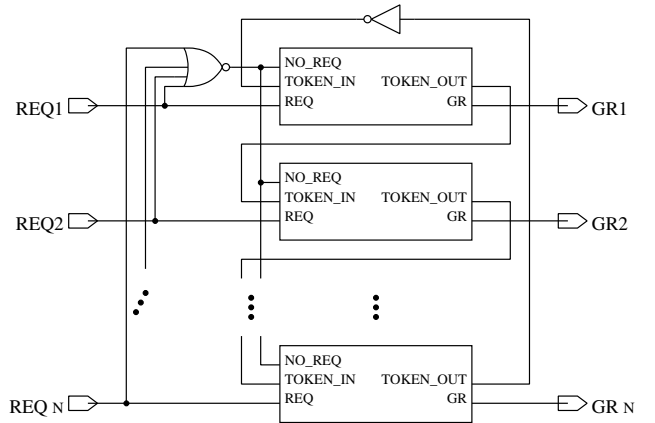


Fig. 4. N-way ring arbiter.

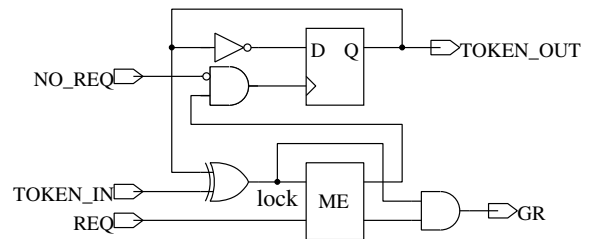


Fig. 5. Ring arbiter module.

The latency and the throughput of this implementation depend on the position of the token in the ring structure.

If a request signal REQ_i arises and its predecessor node in the ring structure has the token, the grant signal GR_i can arise after the token moves to the current node. If the predecessor node has no token, the grant signal cannot arise until the token reaches through the ring. Thus, it can be considered that the average latency is proportional to $N/2$. On the other hand, the number of cascaded MEs is only 1. Therefore, the risk of entering metastable states is small compared with other implementation which has large cascaded MEs. In addition, the proposed lazy ring arbiters are fair since there is no priority in the token-ring while the latency variance is large.

IV. SQUARE FAIR ARBITER

As mentioned above, the triangle mesh arbiter is unfair due to the topological order of primary input signals. This is because the level of one input signal for each two-way ME is different from that of another input signal. The level means the number of MEs which are passed from the primary input. In order to make a fair mesh arbiter, it is necessary to uniform the levels of input signals for each two-way ME.

Figure 6 shows the top-level view of the proposed square arbiter. When the number of inputs N is even, each stage has

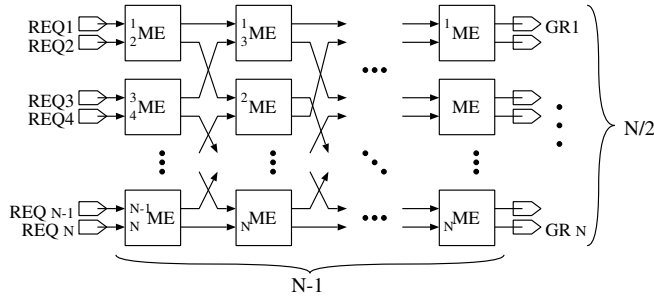


Fig. 6. N -way square arbiter.

$N/2$ MEs and each path from the primary input REQ_i to the primary output GR_i consists of $N - 1$ MEs. It composes a $(N - 1) \times N/2$ square mesh structure. For example, Fig. 7 shows a 6-way square arbiter structure. In this paper, we are not concerned with an algorithm to determine the mesh interconnects. When the number of inputs N is odd, each stage has $\lfloor N/2 \rfloor$ MEs. The number of MEs for each path is $N - 1$. For example, Figure 8 shows a 5-way square arbiter structure. The input signal REQ_6 is deleted from the 6-way square arbiter shown in Fig. 7, resulting in the 5-way square arbiter shown in Fig. 8. It can be said that a $(2n - 1)$ -way square arbiter can be constructed from a $2n$ -way square arbiter by deleting one of request signals.

In the square arbiters, the total number of MEs is $N(N - 1)/2$. The number of cascaded MEs for each path from a request signal REQ_i to its grant signal GR_i is all $N - 1$. Thus, the latency is proportional to N . In addition, the risk of entering metastable states may be large.

V. PERFORMANCE COMPARISON

A. Evaluation Setup

We designed four different N -way ($3 \leq N \leq 16$) arbiters; cascaded tree arbiters, triangle mesh arbiters, the proposed

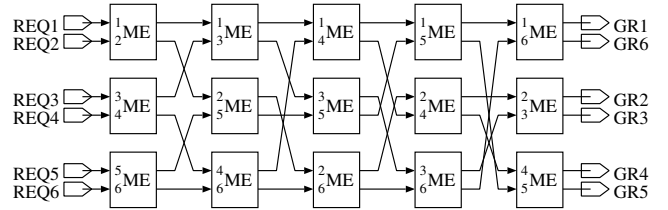


Fig. 7. 6-way square arbiter.

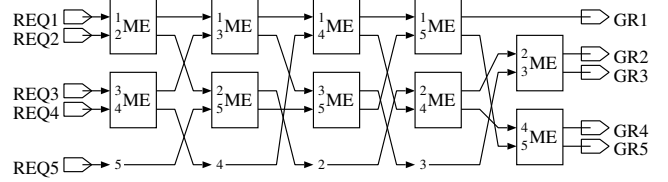


Fig. 8. 5-way square arbiter.

lazy ring arbiters, and the proposed square arbiters. We used 65nm process technology parameters given by a fabrication vendor. We used the Cadence Virtuoso layout editor for an ME cell and a C-element cell design, the Synopsys Design Compiler for technology mapping, the Synopsys Astro for place-and-route, the Cadence Assura for extracting SPICE files from the placed-and-routed data, and the HSPICE analog simulator for performance evaluation, respectively. When we made floorplans of synthesized circuits, we assumed that the utilization of cells is 50%. We assumed that process variation parameters are taken as typical values, the voltage of circuits is 1.1V, and the temperature is 50 degrees Celsius. FO4 inverter delay in this process is about 30(psec).

We first assumed that only one request signal arises. We evaluated latency which means the duration from the time a request signal arises until the time its grant signal arises as shown in the upper side of Fig. 12 (a) for each input. Then, we calculate average values. Second, we assumed

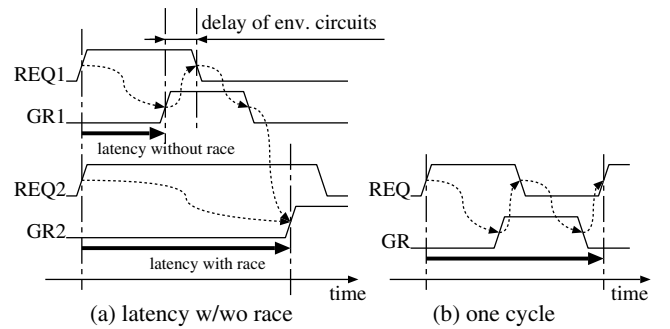


Fig. 12. Latency and cycle without race.

that two request signals arise at the same time. We evaluated latency with race as shown in Fig. 12 (a). Figure 13 shows a simulation environment. As shown in Fig. 13, a grant signal GR_i is inverted rapidly through an INV gate and an AND gate, then issues its request signal REQ_i . We evaluated all the pairs of request signals and calculated average values. As shown in Fig. 12 (a) and Fig. 13, the latency with race includes the delay of environmental circuits, i.e. an INV gate

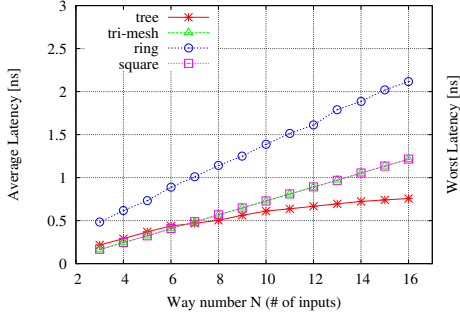


Fig. 9. Average latency without race.

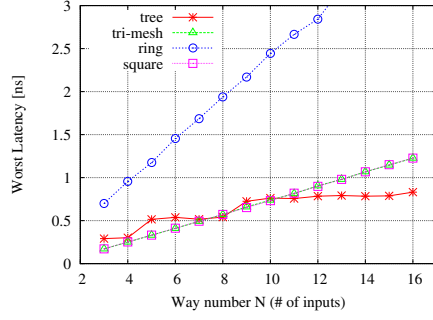


Fig. 10. Worst latency without race.

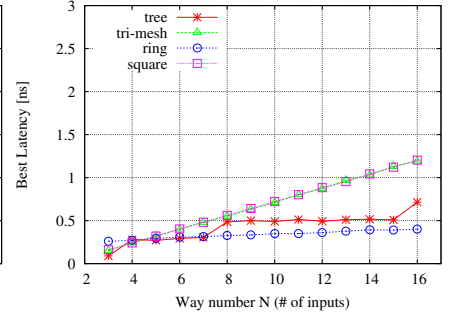


Fig. 11. Best latency without race.

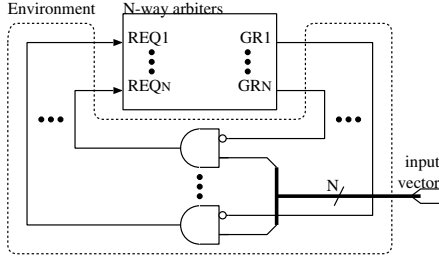


Fig. 13. Simulation environment.

and an AND gate. When we evaluated energy, we assumed that one cycle is the duration from the time one of requests arises to the time the request arises again after its grant signal arises and falls as shown in Fig. 12 (b).

B. Latency Comparison

We first compare latency without race. Figure 9, Fig. 10, and Fig. 11 show the average, worst, and best latency, respectively. The horizontal axis represents the number of inputs N . The vertical axis represents the latency. From Fig. 9, it can be seen that the average latency of the proposed square arbiters is the same as that of the triangle mesh arbiters. The average latencies of the triangle mesh, the square, and the proposed ring arbiters are proportional to the number of inputs N . It can be seen that the latencies of triangle mesh arbiters and the square arbiters are the smallest among four arbiters when the number of inputs is 6 or less, and the latencies of the cascaded tree arbiters are the smallest when the number of inputs is 7 or more. It can be said that the triangle mesh and the square are suitable when the number of inputs is smaller than 6. Then, it can be also said that the cascaded tree arbiters are suitable when the number of inputs is larger than 7. From Fig. 10 and Fig. 11, it can be seen that the worst and best latencies of the cascaded tree arbiters depend on the worst level and best level from the root node, i.e. they are proportional to $\lceil \log_2 N \rceil, \lfloor \log_2 N \rfloor$, respectively. From Fig. 11, it can be seen that the best latency of the proposed ring arbiters is almost equal since a node can issue its grant signal immediately when the token exists in the predecessor node. From Fig. 9 ~ Fig. 11, it can be seen that the latency variance of the triangle mesh arbiters and square arbiters is very small and the latency variance of the ring arbiters is very large.

Then, we compare latency with race as shown in Fig. 12 (a). Figure 14, Fig. 15, and Fig. 16 show the average, worst, and best latency, respectively. From Fig. 14, it can be seen that the average latencies of the triangle mesh arbiters are the smallest when the number of inputs is 11 or less, and the latencies of the cascaded tree arbiters are the smallest when the number of inputs is 12 or more. Compared with Fig. 9, it can be seen that the average latencies of the tree arbiters, the triangle mesh arbiters, and the square arbiters are about twice as large as the latencies without race. On the other hand, the average latencies of the proposed ring arbiters is smaller than twice of the latencies without race. This is because the latency of the ring arbiters depends on the number of active requests in the ring structure. Then, from Fig. 15 and Fig. 16, it can be also seen that the worst and best latencies of the cascaded tree arbiters depend on the worst level and best level from the root node, i.e. they are proportional to $\lceil \log_2 N \rceil, \lfloor \log_2 N \rfloor$, respectively. It can be seen that the latency variance of the ring arbiters is very large since the latency depends on the position of the token. Then, it can be also seen that the latency variance of the proposed square arbiters is small since they have no priority. On the contrary, the latency variance of the triangle mesh arbiters is larger than that of the square arbiters since they have priority. As a result, it can be said that the proposed square arbiters are suitable for fair arbitration.

C. Throughput Comparison

We compare the throughput of four different N -way arbiters. In this paper, we define that throughput is the reciprocal number of the average interval between grant signals.

Figure 17 shows the maximum throughput. The horizontal axis represents the number of inputs N . The vertical axis represents the throughput. From Fig. 17, it can be seen that the maximum throughput of the mesh arbiters is inversely proportional to the number of inputs N . On the other hand, the maximum throughput of the proposed ring arbiter does not depend on the number of inputs N . This is because the ring arbiters can issue grant signals continuously with moving the token, when all the requests arise. Then, from Fig. 18, it can be seen that the minimum throughput of the triangle mesh arbiters, the proposed square arbiters, and the proposed ring arbiters is inversely proportional to the number of inputs N .

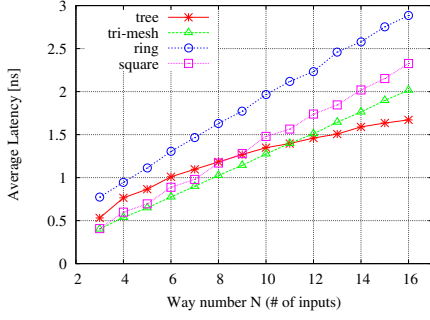


Fig. 14. Average latency with race.

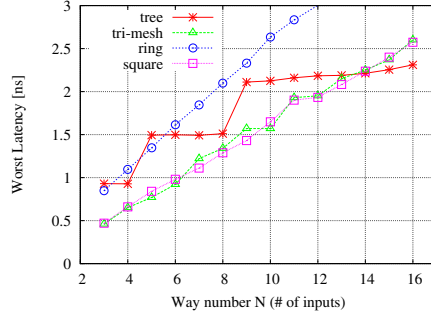


Fig. 15. Worst latency with race.

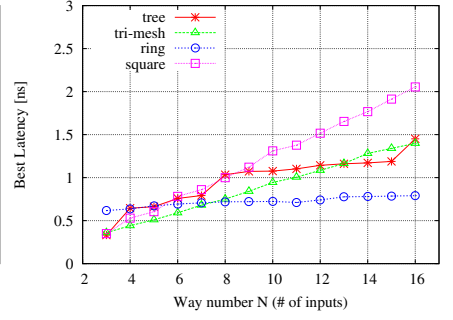


Fig. 16. Best latency with race.

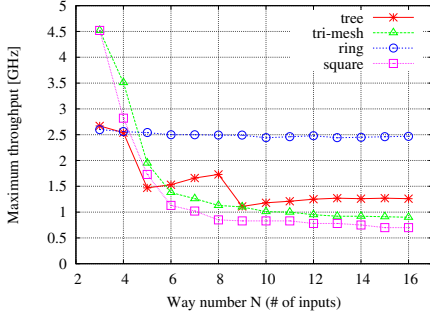


Fig. 17. Maximum throughput.

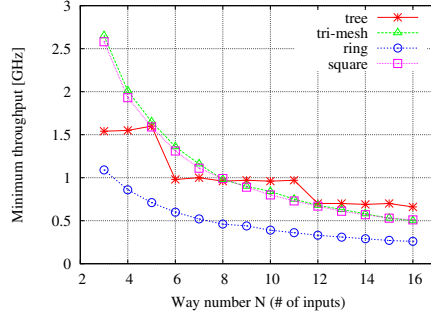


Fig. 18. Minimum throughput.

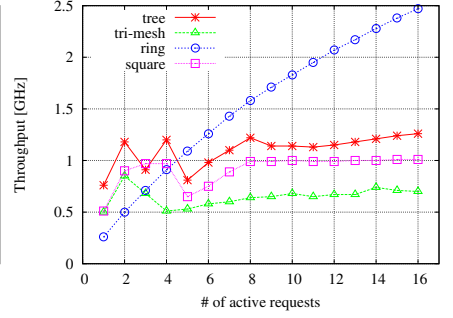


Fig. 19. Throughput of 16-way arbiters.

As mentioned above, the throughput of the ring arbiters depends on the number of active requests in the ring structure. Figure 19 shows the throughput of 16-way arbiters when the number of active requests is varied from 1 to 16. From Fig. 19, it can be seen that the throughput of the ring arbiters increases as the number of active requests increases. On the other hand, the throughput of the tree arbiters, the triangle mesh arbiters, and the square arbiters is almost the same when the number of active request is larger than 8. Thus, in the viewpoint of throughput, it can be said that the ring arbiters are suitable when the number of active requests is 5 or more, the proposed square arbiters and the triangle mesh arbiters are suitable when the number of active requests is 4 or less. In addition, the throughput strongly depends on the delay of environmental circuits. If the environmental delay is large, the delay of the token ring can be concealed. Thus, the ring arbiters may be also suitable for small active request number.

D. Area and Energy Comparison

Figure 20 shows the area of four different arbiters. The horizontal axis represents the number of inputs N . The vertical axis represents the area. From Fig. 20, it can be seen that the area of the triangle mesh arbiters and the proposed square arbiters increases exponentially. It can be also seen that the area of the tree arbiters and the proposed ring arbiters is proportional to the number of inputs N . Thus, in the viewpoint of area, it can be said that the tree arbiters are suitable when the number of inputs is 6 or more, the proposed square arbiters and the triangle mesh arbiters are suitable when the number of inputs is 5 or less.

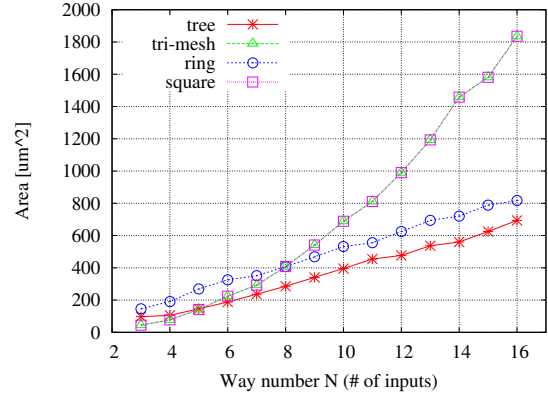


Fig. 20. Area.

Figure 21 shows the average energy without race. The horizontal axis represents the number of inputs N . The vertical axis represents the average energy of one cycle (Fig. 12 (b)). Figure 22 shows the average energy under the maximum throughput condition.

From Fig. 21, it can be seen that the average energy of the triangle mesh, the square, and the ring arbiters is proportional to the number of inputs N . It can be seen that the average energy of ring arbiters is about three times larger than those of other arbiters for a light work load environment. On the other hand, for a heavy work load environment, the average energy of ring arbiters is the smallest among four type arbiters when the number of inputs N is larger than 5. Consequently, it can be said that the triangle mesh and the square arbiters are suitable when the number of inputs is small in the viewpoint

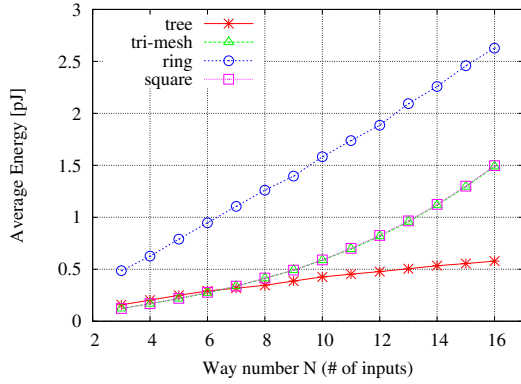


Fig. 21. Average energy-per-cycle without race.

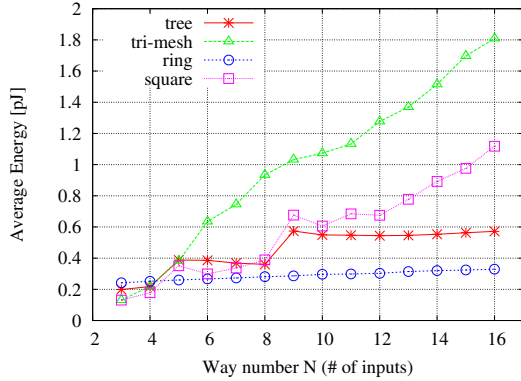


Fig. 22. Average energy-per-cycle under the maximum throughput condition.

of energy. When the number of inputs is large, the ring arbiters are suitable for the heavy work load environment, and the tree arbiters are suitable for the light work load environment, respectively.

E. Comprehensive Discussion

It follows from what has been said thus far that we should select the appropriate arbiters in accordance with the required specification. Table I and Table II show summaries of four type arbiters for large input number and small input number, respectively.

It can be concluded that the proposed square arbiters are suitable when the number of inputs N is 5 or less since all

TABLE I
FEATURES OF FOUR ARBITERS FOR LARGE N ($N \geq 6$).

	tree	tri-mesh	ring	square
Latency	good	bad	bad	bad
Throughput	bad	bad	good	bad
Area	good	bad	good	bad
Energy	good	good	bad	good

TABLE II
FEATURES OF FOUR ARBITERS FOR SMALL N ($N \leq 5$).

	tree	tri-mesh	ring	square
Latency	bad	good (unfair)	bad	good
Throughput	bad	good	bad	good
Area	good	good	good	good
Energy	good	good	bad	good

the features of square arbiters are better than others. In the viewpoint of latency, the tree arbiters are suitable when the number of inputs N is 12 or more, the square arbiters are suitable when the number of inputs N is 11 or less. Then, in the viewpoint of throughput, the ring arbiters are suitable if the number of input N is larger than 5 and the number of active requests is large. In the viewpoint of area and energy, the tree arbiters are suitable. In addition, in the viewpoint of the risk of entering metastable states, the ring arbiters are suitable since the number of cascaded MEs is only 1 as shown in Table III.

TABLE III

THE NUMBER OF CASCADED MES.

	tree	tri-mesh	ring	square
Cascaded MEs	$\log_2 N$	$N - 1$	1	$N - 1$

VI. CONCLUSION

Arbitration circuits play an important role to manage shared resources. In this paper, two new arbiter circuits have been proposed. One circuit is based on the token-ring arbiters and another circuit is based on the mesh arbiters. The proposed lazy ring arbiters can achieve high throughput arbitration for a heavy work load environment. The proposed square arbiters and lazy ring arbiters can issue grant signals fairly. We have designed four different arbiters using 65nm process technologies and compare their performance. As the result, it can be concluded that the proposed square arbiter is suitable when the number of inputs is 5 or less. Then, it can be concluded that we must select the appropriate arbiters considering trade-off between latency, throughput, area, and energy when the number of inputs is larger than 5.

ACKNOWLEDGMENTS

This work was supported in part by CREST(Core Research for Evolutional Science and Technology) of Japan Science and Technology Corporation, Japan Society for the Promotion of Science Grant-in-Aid for Scientific Research (19300009) and by VLSI Design and Education Center (VDEC), the University of Tokyo in collaboration with Synopsys, Inc and Cadence Design Systems, Inc

REFERENCES

- [1] D. J. Kinniment, *Synchronization and Arbitration in Digital Systems*. John Wiley & Sons, Ltd, 2007.
- [2] W. W. Plummer, "Asynchronous arbiters," *IEEE Transactions on Computers*, vol. 21, no. 1, pp. 37–42, Jan. 1972.
- [3] D. L. Black, "On the existence of delay-insensitive fair arbiters: Trace theory and its limitations," *Distributed Computing*, vol. 1, pp. 205–225, 1986.
- [4] K. S. Low and A. Yakovlev, "Token ring arbiters: An exercise in asynchronous logic design with petri nets," *Technical Report No.537, Dept of Comp. Sci., University of Newcastle upon Tyne*, Nov. 1995.
- [5] M. B. Josephs and J. T. Yantchev, "CMOS design of the tree arbiter element," *IEEE Transactions on VLSI Systems*, vol. 4, no. 4, pp. 472–476, Dec. 1996.
- [6] A. Bystrov and A. Yakovlev, "Ordered arbiters," *IET Electronics Letters*, 27th, vol. 35, no. 11, pp. 877–879, May 1999.
- [7] A. Bystrov, D. J. Kinniment, and A. Yakovlev, "Priority arbiters," in *Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems*. IEEE Computer Society Press, pp. 128–137, Apr. 2000.