High Performance Two-Select Arbiter

Maher Abdelrasoul[1], Mohammed Sayed[1], Victor Goulart[1,2]

[1]ECE department, Egypt-Japan university of Science and Technology (E-JUST), Egypt

[2]Center for Japan-Egypt Cooperation in Science and Technology, Kyushu University, Japan

email: {maher.salem, mohammed.sayed}@ejust.edu.eg, victor.goulart@acm.org

Abstract—**Arbitration circuit plays an important role in defining the system performance and latency in systems having shared resources. In this paper, we focused on the problem of granting two requesters simultaneously. We proposed a new solution to fix a bug of three-Dimentional programmable two-select circuit, which is the fastest arbitration circuit in the literature. We implement all architectures concerning the two-select arbitration on 65 nm CMOS technology. Our circuit showed the smallest delay and, in average, showed the second smallest area among all analogue circuits.**

I. INTODUCTION

Arbiters are usually found in systems that have shared resources with many requesters. Arbiters are used to control access to those resources. Mostly it is preferred to use strong fairness arbiters to avoid starvation at any requester. The most widely known arbiter is Round Robin Arbiter (RRA), which is simple in design, fast, and has strong fairness [1]. There are many researches on arbiter design aiming to have fast arbitration circuits. Most of them consider that the shared resource cannot accept access from more than one requester [2]. However, some techniques accept more than one requester to access the same resource at the same time with more control. In our work, we focus on two-select arbiter, which controls the granting of not more than two requesters among many requesters. However, the architectures presented in this paper, with little modification, can be used in multi-select arbiter for systems having availability to access many resources at the same time. Our architecture shows the best performance over the other working architectures we are aware of.

II. RELATED WORK

In the state-of-art, only two researches targeted two-select RRAs. The first one is 3-Dimentional programmable 2-select (3DP2S) arbiter. 3DP2S was proposed in [3]. The arbitration circuit of 8-point 3DP2S is shown in Fig. 1. It consists of log2(8) stages of unit blocks (UBs) and edge detector stage. The UB of 3DP2S is a thermometer-coded adder saturated at 2 i.e. whenever the sum of the two inputs is larger than 2 the output would be 2. Further, the UB also takes two pointer bits (i.e. inputs priorities) and outputs the OR result of them. The pointer bits control the adder function. If the right input priority is logic high, the UB will not add its inputs and pass the right input as it is. Therefore, the result of additions will propagate through stages to the paths of requesters that have lower priority. The result of the three stages is divided into two vectors; the left bits, and the right bits. Finally, the edge detector is used to detect the zero-one transition. The detected one at the first vector represents the first active requester, while the detected one at the second vector represents the second active requester. The main problem in this algorithm occurs when the highest priority requester is active. The sequence of additions results in a vector of ones. This vector when passed to the edge detector stage results a vector of zeros. This bug was not mentioned in the original paper [3].

The bug in the 3DP2S architecture was fixed in [4]. We will name this circuit 3DP2S_OZU. Its idea is based on ANDing the priority vector with input request vector to indicate whether the highest priority requester is active or not. If it is active, the first grant vector will be the priority vector itself where the priority vector grants the highest priority requester. On the other hand, if the highest priority requester is not active, the grant vector generated by the circuit will be activated. The control is done through a multiplexer circuit. The fixed circuit is shown in Fig. 2.

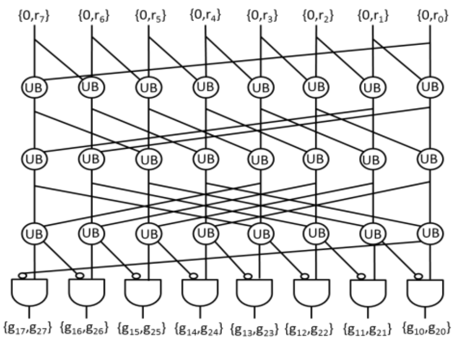

Fig. 1: 8-point 3DP2S arbitration circuit.

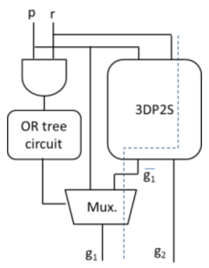

Fig. 2: 3DP2S_Ozu modified arbitration circuit.

Another two-select arbiter, named RRA-2pick-Ozu, was proposed in [4]. Its architecture is based on fixed priority encoder (FPE) idea [5]. FPE picks the first two active requesters in its input. It was shown that RRA-

2pick-Ozu architecture has a significant reduction in area, where the 3DP2S architecture has higher performance for arbiter having a number of requesters lower than 32.

III. Proposed Solution

We have worked on the 3DP2S to solve its bug in a more efficient way to improve its performance. Our idea is also based on ANDing the priority vector with input request vector to indicate whether the highest priority requester is active or not. However, our circuit is different. Our circuit ANDs the priority vector with the request vector then, ORing the result with the grant vector which is generated by the arbitration circuit. Our circuit is shown in Fig. 3. Comparing our circuit, named as 3DP2S_E with the 3DP2S_Ozu, we expect our circuit to improve both time and area. In terms of area, we have replaced the OR tree and the multiplexer circuits by a bit-wise OR circuit. Figs. 2 and 3 show the critical paths of the two circuits as dashed lines. Our circuit just increased the critical path of the 3DP2S circuit by one OR logic gate. On the other hand, in 3DP2S_Ozu the critical path increased by a multiplexer. In terms of area, both of our circuit and 3DP2S_Ozu circuit have a bitwise ANDing circuit. However, our circuit uses a bitwise ORing circuit instead of the OR tree and the multiplexer circuits used in 3DP2S_Ozu. In next section, a complete evaluation study and numerical results are presented.

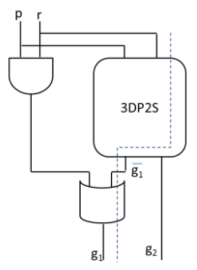

Fig. 3: 3DP2S arbitration circuit with our modification.

IV. Evaluation and Results

We evaluated the performance of the proposed 3DP2S_E architecture versus both 3DP2S_Ozu and RRA_2pick_Ozu architectures for arbiter sizes starting from 4 to 32 in terms of delay and area. We described the three architectures in VHDL. We verified the correctness of each description by simulating it using ISim tool of Xilinx 14.5. Synthesis has been done using Cadence Encounter RTL compiler RC11.10 with TSMC 65 nm technology. The delay and area results are shown in Figs. 4 and 5 respectively. In terms of delay, RRA_2pick_Ozu circuit shows a high delay for all sizes of the arbiter except for 32 point arbiter. Therefore, if we keep the competition between 3DP2S_E and 3DP2S_Ozu, our design (3DP2S_E) shows the smallest delay with average reduction 5% compared to the 3DP2S_Ozu circuit. Regarding the area, although RRA_2pick_Ozu circuit still has the advantage of the smallest area, our circuit as expected shows an average reduction of 4% in area than the 3DP2S_Ozu.

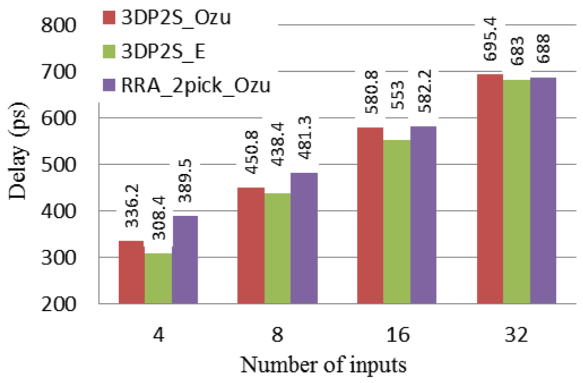

Fig. 4: delay results

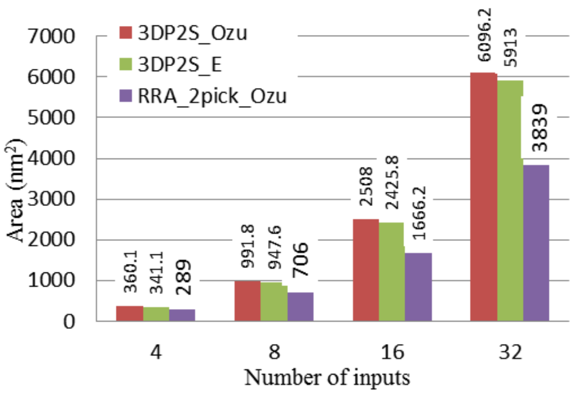

Fig. 5: area results

V. Conclusions

In this paper, we presented a complete literature review on the problem of granting two requesters at the same cycle time. We focused on a bug in the circuit with the smallest delay for arbiters that is used with low number of requesters. Further, we reviewed the only solution that was presented in the literature. We proposed a new solution that solves the bug and implemented the design on ASIC CMOS technology. Our proposed design shows the best results in terms of delay for arbiter sizes starting from 4 to 32. Besides, it has smaller area than other solutions.

References

[1] W. J. Dally and B. Towles, “Route Packets, Not Wires: On-Chip Inteconnection Networks,” In Proc. of 38th Design Automation Conference (DAC), pp.684-689, 2001
[2] M. Abdelrasoul, M. Ragab, V. Goulart, "Impact of Round Robin Arbiters on router's performance for NoCs on FPGAs," IEEE International Conference on Circuits and Systems (ICCAS), pp.59-64, 2013
[3] J. S. Ahn, D. K. Jeong, and S. Kim, “Fast three-dimensional programmable two-selector”, Electronics Letters, vol. 40, no. 18, 2004
[4] H.F. Ugurdag, F. Temizkan, O. Baskirt, and B. Yuce, “Fast two-pick n2n round-robin arbiter circuit”, Electronics Letters, vol. 48, vo. 13, 2012
[5] P. Gupta and N. McKeown, “Designing and implementing a fast crossbar,” Micro IEEE, vol.19, Issue 1, pp.20-28, 1999