We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

185,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



20

Arbitration Schemes for Multiprocessor Shared Bus

Dr. Preeti Bajaj and Dinesh Padole
G.H. Raisoni College of Engineering, Nagpur
India

1. Introduction

Performance of Multicore Shared bus Embedded Controller depends on how effectively the sharing resources can be utilized. Common bus in System on Chip is one of the sharing resources, shared by the multiple master cores and also acting as a channel between master core and slave core (peripherals) or Memories. Arbiter is an authority to use the shared resource (Shared bus) effectively, so performance also depends on arbitration techniques.
The arbitration mechanism is used to ensure that only one master has access to the bus at any one time. The arbiter performs this function by observing a number of different requests to use the bus. Master may request to bus master (arbiter) to use the bus during any cycle. The arbiter will sample the request on the rising of the clock and then use predefined algorithm to decide which master will be the next to gain access to the bus. On-chip communication architecture plays an important role in determining the overall performance of the System-on-Chip (SoC) design. In the recourse sharing mechanism of SoC, the communication architecture should be flexible to offer high performance over a wide range of data traffic.

2. Arbitration techniques.

There are several arbitration techniques has been developed mention as below.

2.1 Static fixed priority algorithm

Static fixed priority is a common scheduling mechanism on most common buses. In a static fixed priority scheduling policy, each master is assigned a fixed priority value. When several masters request simultaneously, the master with the highest priority will be granted. The advantage of this arbitration is its simple implement and small area cost. The static priority based architecture does not provide a means for controlling the fraction of communication bandwidth assigned to a component. If masters with high priority requests frequently, it will lead to the starvation of the ones with low priority.
Advantages: It is simple in implement & Small area cost
Disadvantages: In Heavy communication traffic, master that has low priority value can not get a grant signal.

2.2 TDM/Round-Robin algorithm

Time division multiplexed (TDM) scheduling divides execution time on the bus into time slots and allocates the time slots to adapters requesting use of the bus. Each time slot can span several physical transactions on the bus. A request for use of the bus might require multiple slot times to perform all required transfers. However, in this architecture, the components are provided access to the communication channel in an interleaved manager, using a two level arbitration protocol.

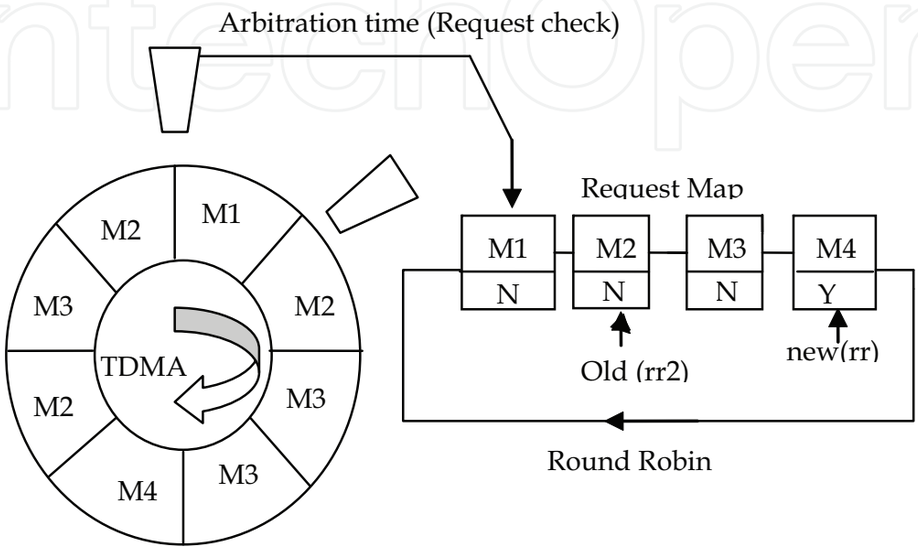


Fig. 1. Round Robin based arbiter communication architecture

The first level of arbitration uses a timing wheel where each slot is statically reserved for a unique master. In a single rotation of the wheel, a master that has reserved more than one slot is potentially granted access to the channel multiple times. If the master interface associated with the current slot has an outstanding request, a single word transfer is granted, and the timing wheel is rotated by one slot. To alleviate the problem of wasted slots, a second level of arbitration is supported. The policy is to keep track of the last master interface to be granted access via the second level of arbitration, and issue a grant to the next requesting master in a round-robin fashion, at figure 1, the current slot is reserved for M1, but it has no data to communicate. The second level increments a round-robin pointer rr2 from its current position at M2 to the next outstanding request at M4.

Advantages: Easy to implement

Disadvantages: Leads to the mistake of data transfer

However, these techniques are often inadequate. In the former, low priority components may suffer from starvation, while high priority components may have large latency. Low system performance because of bus distribution latency in a bus cycle time. Hence there is need to design some more efficient arbitration scheme. The chapter presents four arbitration schemes for system on chip communication as below.

- Static Lottery Bus architecture
- Dynamic lottery bus architecture
- ATM switch architecture
- Fuzzy Logic based arbiter

2.3 Static Lottery Bus architecture

The core of the LOTTERYBUS architecture is a probabilistic arbitration algorithm implemented in a centralized "lottery manager" for each bus in the communication architecture. The architecture does not presume any fixed communication topology. Hence, various SoC components may be interconnected by an arbitrary network of shared channels or a flat system wide bus as shown in figure 2.

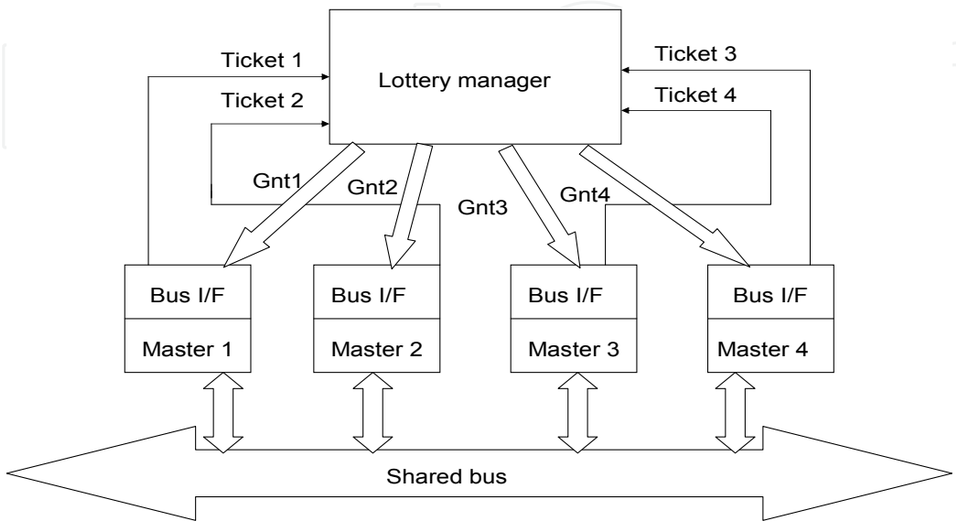


Fig. 2. Lottery manager for a bus in a Lottery bus based communication architecture

The lottery manager accumulates requests for ownership of the bus from one or more masters, each of which is (statically) assigned a number of "lottery tickets," as shown in figure 3. The manager pseudo-randomly chooses one of the contending masters to be the winner of the lottery, favoring masters that have a larger number of tickets, and grants access to the chosen master for a certain number of bus cycles. Multiple word requests may be allowed to complete without incurring the overhead of a lottery drawing for each bus word. However, to prevent a master from monopolizing the bus, a maximum transfer size is used to limit the number of bus cycles for which the granted master can utilize the bus Also, the architecture pipelines lottery manager operations with actual data transfers, to minimize idle bus cycles. The inputs to the lottery manager are a set of requests (one per master) and the number of tickets held by each master. The output is a set of grant lines (again one per master) that indicate the number of words that the currently chosen master is allowed to transfer across the bus. The arbitration decision is based on a lottery. The lottery manager periodically (typically, once every bus cycle) polls the incoming request lines to see if there are any pending requests. If there is only one request, a trivial lottery results in granting the bus to the requesting master. If there are two or more pending requests, then the master to be granted access is chosen using the approach described next.

2.3.1 Lottery-based arbitration algorithm

Let the set of bus masters be C1, C2, C3, C4 & Let the number of tickets held by each master are t1, t2, t3, t4. At any bus cycle, let the set of pending bus access requests be represented by a set of Boolean variables r_i (i=1, 2…, n) where r_i=1 if component C_i has a pending request, and r_i=0 otherwise. The master to be granted is chosen in a pseudo-random way, favoring components with larger numbers of tickets. The probability of granting component C_i is given by

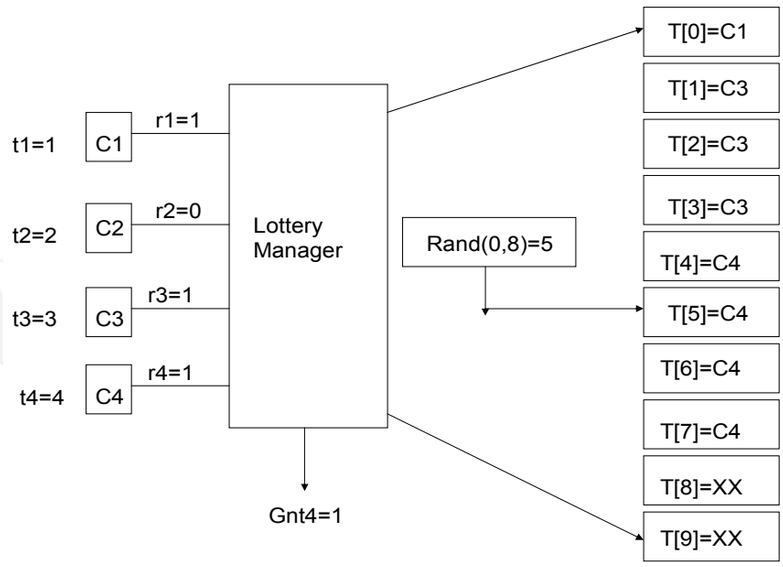


Fig. 3. Illustration of a lottery that determines which master should be awarded ownership of the bus.

$$P(C_i) = \frac{r_i * t_i}{\sum_{j-1}^{n} r_j * t_j}$$

Figure 3 shows an example where three out of four bus masters have contending requests, with tickets in the ratio 1:3:4. For this request map and ticket holding combination, the lottery-based approach should result in P(C1)=0.12, P(C2)=0, P(C3)=0.37, P(C4)=0.5, To make an arbitration decision, the lottery manager examines the number of "active" tickets, or the number of tickets in possession of the set of components that have pending requests. This is given by

$$\sum_{j=1}^{n} r_j * t_j$$

It then generates a pseudo-random number (or picks a winning "ticket") from the range $\left[0, \sum_{j=1}^{n} r_j * t_j\right]$ to determine which component to grant the bus. If the number falls in the range [0,r1*t1]the bus is granted to component C1, if it falls in the range [r1*t1,r1*t1+r2t2] it is granted to component C2. and so on. In general, if it lies in the range $\left[\sum_{k=1}^{i} r_k * t_k, \sum_{k=1}^{i+1} r_k * t_k\right]$ it is granted to component C_{i+1} The component with the largest number of tickets occupies the largest fraction of the total range, and is consequently the most likely candidate to receive the grant, provided the random numbers are uniformly distributed over the interval 1 $\left[0, \sum_{j=1}^{n} r_j * t_j\right]$ For example, in Figure 3, components C1, C2, C3 and C4 are

assigned 1, 2, 3, and 4 tickets, respectively. However, at the instant shown, only C1,C3,C4 have pending requests hence the number of current tickets is

$$\sum_{j=1}^{n} r_j * t_j = 1 + 3 + 4 = 8$$

Therefore, a random number is generated uniformly in the range (0, 8) In the example, the generated random number is 5, and lies between r0*t0+r1*t1+r2*t2=4 and r0*t0+r1*t1+r2*t2+r3*t3=8. Therefore, it indexes to a ticket owned by component C4. According to the rule described above, and as illustrated in Figure 3. Therefore, the bus is granted to component C4 win the very first lottery.

Figure 4 shows block diagram of Lottery Bus architecture. It contains three basic blocks. (1)Lottery manager:-In this block r1, r2, r3, r4 are the requests signal of the master and t1, t2, t3 and t4 are the tickets of the master respectively. That will generate the ticket values that are r1t1, r1t1+r2t2, r1t1+r2t2+r3t3, r1t1+r2t2+r3t3+r4t4. (2)Random number generator:- Random number generator is working on the principle of pseudo random binary sequence generator .That will generate the number randomly. (3) Comparison and grant generation hardware:- The random number is compared in parallel against all four partial sums. Each comparator outputs a "1" if the random number is less than the partial sum at the other input. Since for the same number, multiple comparators may output a "1" (e.g., if r1=1 and the generated random number is smaller than, all the comparators will emit "1"), it is necessary to choose the first one, starting with the first comparator. For example, for the request map 1011 if the generated random number is 5, only's C4 associated comparator will output a "1." However, if the generated random number is "1," then all the comparators will output a "1," but the winner is C1. The architecture is model using VHDL for three masters. Ticket values are keeping fixed. Figure 4 shows the simulation results for the discussed architecture. Here t0, t1,t2 & t3 are tickets values and gnt0,gnt1,gnt2 & gnt3 are grant signals of the master processor. Signal n1 is random number generated signal and signal h0, h1, h2 &h3 are calculated value for the master or processor according to it's ticket value and request signal r.

As shown in figure 4 the signal r(0), r(1) ,r(2) and r(3) are the request of master 0,master 1,master 2 and master 3 respectively the signal t0, t1 , t2 and t3 are the ticket values of master 0,master 1,master 2 and master 3 respectively. The signal s0, s1, s2 and s3 are the total ticket values of master 0, master 1, master 2 and master 3 respectively. The signal n1 represents the number generated by pseudo random number binary sequence generator (figure 4). The signal gnt0, gnt1, gnt2 and gnt3 are the grant signal of master 0, master 1, master 2 and master 3 respectively. Figure 4 shows the simulation results for static lottery bus as per the algorithm. The numbers in the simulation results indicate the number of master getting the grant of shared bus utilization.

2.4 Dynamic lottery bus architecture

In this architecture (figure 5), the inputs to the lottery manager consist of a set of request lines (r0r1r2r3), and the number of tickets currently possessed by each corresponding master that are generated by ticket generated by ticket generator. Therefore, under this architecture, not only Range of current tickets varies dynamically but it can take on any arbitrary value (unlike the static case, where it was fixed). Therefore at each lottery, the lottery manager

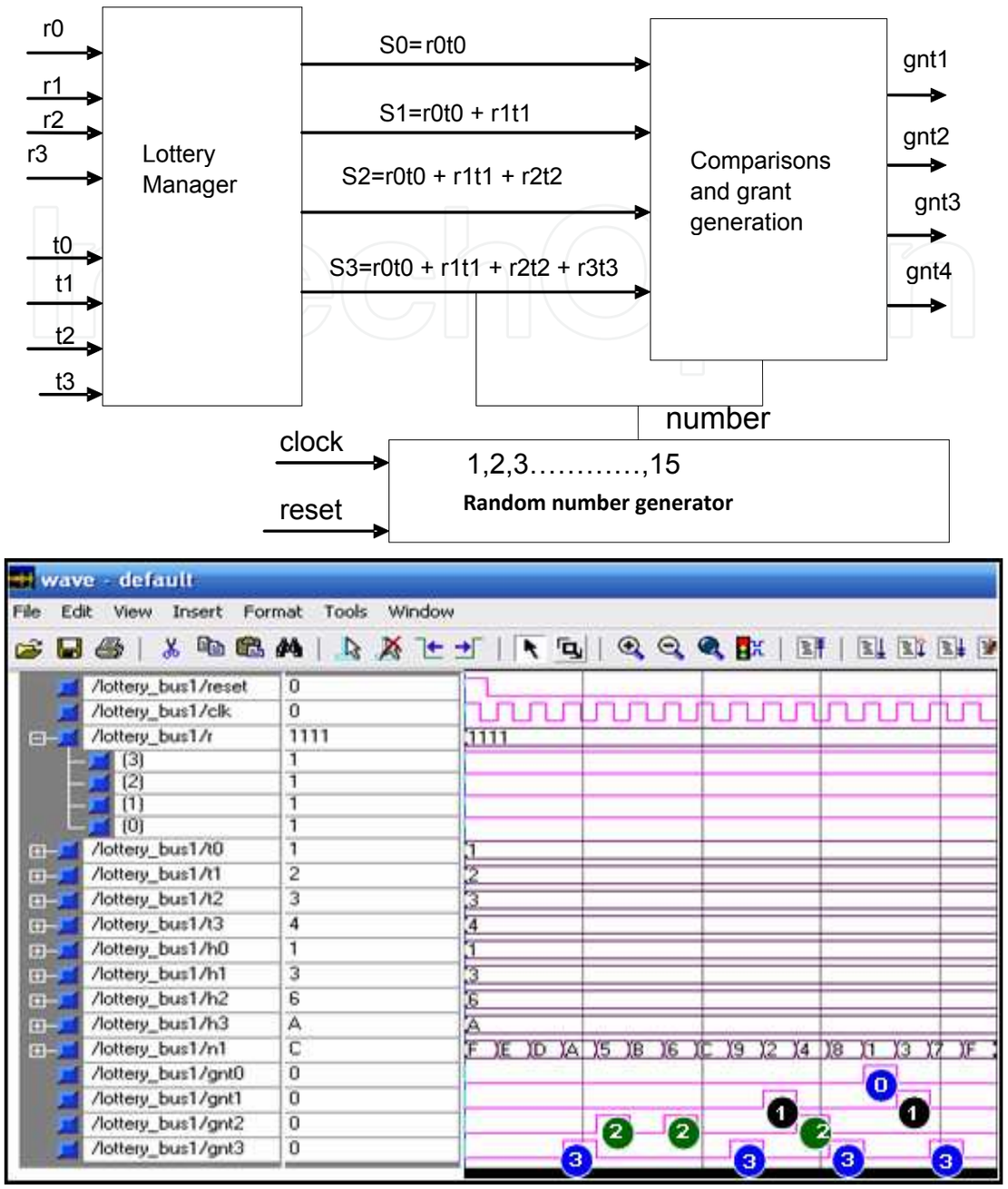


Fig. 4. Structure and simulation results of Static Lottery Based arbiter

needs to calculate for each component C_i, the partial sum $\sum_{j=1}^{n} r_j * t_j$. This is implemented using a bit wise AND operation and tree of adder, as shown in figure 5. The final result, T=r0t0+r1t1+r2t2+r3t3, defines the range in which the random number must lie. A limitation of this implementation is that distribution of the resulting random number is not uniform. The rest of the architecture consists of comparison and grant hardware, and follows directly from the design of the static lottery manager.

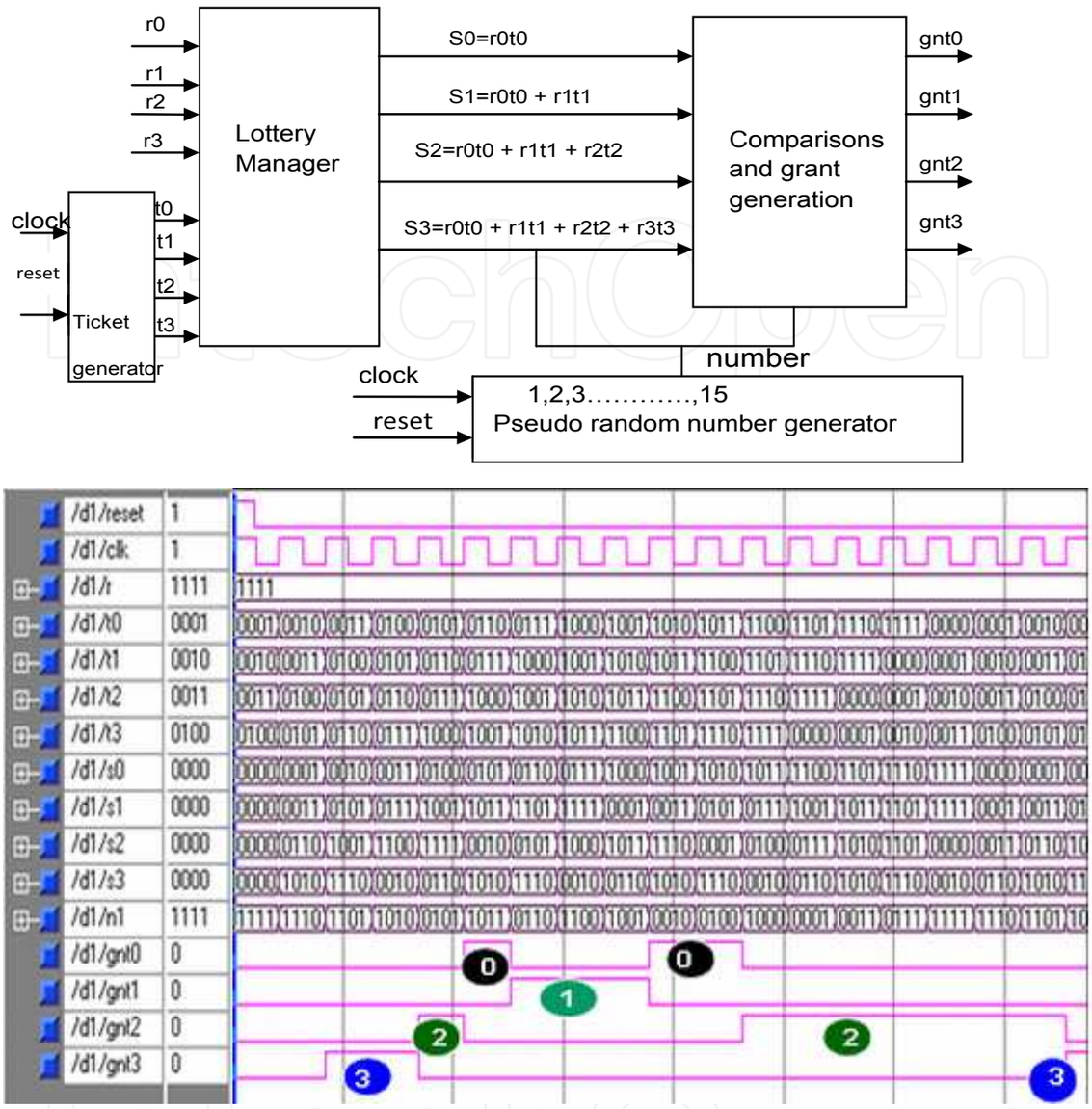


Fig. 5. Structure & simulation Results of Dynamic Lottery bus

The architecture is modeled using VHDL. Ticket values are keeping varying. Figure 5 shows the waveforms for the discussed architecture. Here t0, t1, t2 & t3 are tickets values and gnt0, gnt1, gnt2 and gnt3 are grant signals of the master processor. Signal n1 is random number generated signal and signal s0, s1, s2 and s3 are calculated value for the master or processor according to its ticket value and request signal r.

Advantages: All the masters that are requesting gain the control of bus.

Disadvantages: If the pseudo random number is greater than total ticket value then none of the masters will get the grant signal.

2.5 ATM switch architecture

In this arbitration algorithm, it accepts three parameters (Requests, Tickets, Adaptive signal) for the input of arbiter. Request and Ticket are the input for the static bus distribution.

Adaptive signal value is used as an additional input to improve the probability of the bus grant. This adaptive signal value is transmitted from the master that requires the bus grant more than another master because of the stressful traffic. Since we do not know which IP is used for the shared bus in advance of the SOC design, the adaptive signal can be fixed by the specific parameter. The master counts the buffer position storing the ATM cell and if the data approaches to the limited amount, the adaptive signal is generated to improve the drawing probability.

$$P(C_i) = \frac{r_i(t_i + a_i)}{\sum_{j=1}^{n} r_j * (t_j + a_j)}$$

Above equation shows the shared bus probability for each master. The current pending request and ticket value is used to obtain the shared probability of each C_i. In order to improve the probability of the master, a_i values are obtained from the look up table and two of the master requests accomplish the bit-wise AND operation by the values i. 'a' is the additional ticket value to solve the problem that if the total ticket value is lower than the pseudo random value, the bus is assigned to the master of the low priority by the priority inversion.

If the pseudo random value is bigger than $\sum_{j=1}^{n} r_j * t_j = 4$, the control signal of MUX generates the enable signal by the OR operation of the request bit from the master. The partial summation value of each master is obtained by the bit-wise AND operation between the request values and the ticket value. If the pseudo random value from LFSR and the total ticket value generate modulo $\left(R, \sum_{j=1}^{n} r_j * t_j \right) = 5$. C4 is assigned to be use because the pending request value is 0001.

In figure 6 (in simulation results) signal r represents for the masters request signal. For the testing all masters accept master 2, are requesting for bus. Signal gnt0 to gnt3 are the grant signal. The master grant signals are indicated by numbers.

Advantages: The adaptive signal is used to solve the problem that the characteristics of LFSR are disappeared if the pseudo random number is bigger than total ticket value.

2.6 Fuzzy logic arbiter

Fuzzy logic has already proved to be an innovative and successful design methodology in certain key areas of embedded control where its attributes of simplicity, sensitivity, robustness and easy optimization are tremendously advantageous. Fuzzy logic has been applied widely across the consumer market, where superior product performance has been achieved whilst reducing development time. Typical "fuzzy goods" that have been particularly successful include control systems in washing machines, air conditioners, cameras and camcorders incorporating an auto focusing mechanism, video cassette recorders and audio systems.

The basic concept of fuzzy sets is a generalization of the classical or crisp set. The crisp set is defined in such a way as to dichotomize the individuals in some given universe of discourse into two groups: members (those that certainly belong in the set) and nonmembers (those that certainly do not). A sharp, unambiguous distinction exists between the members and

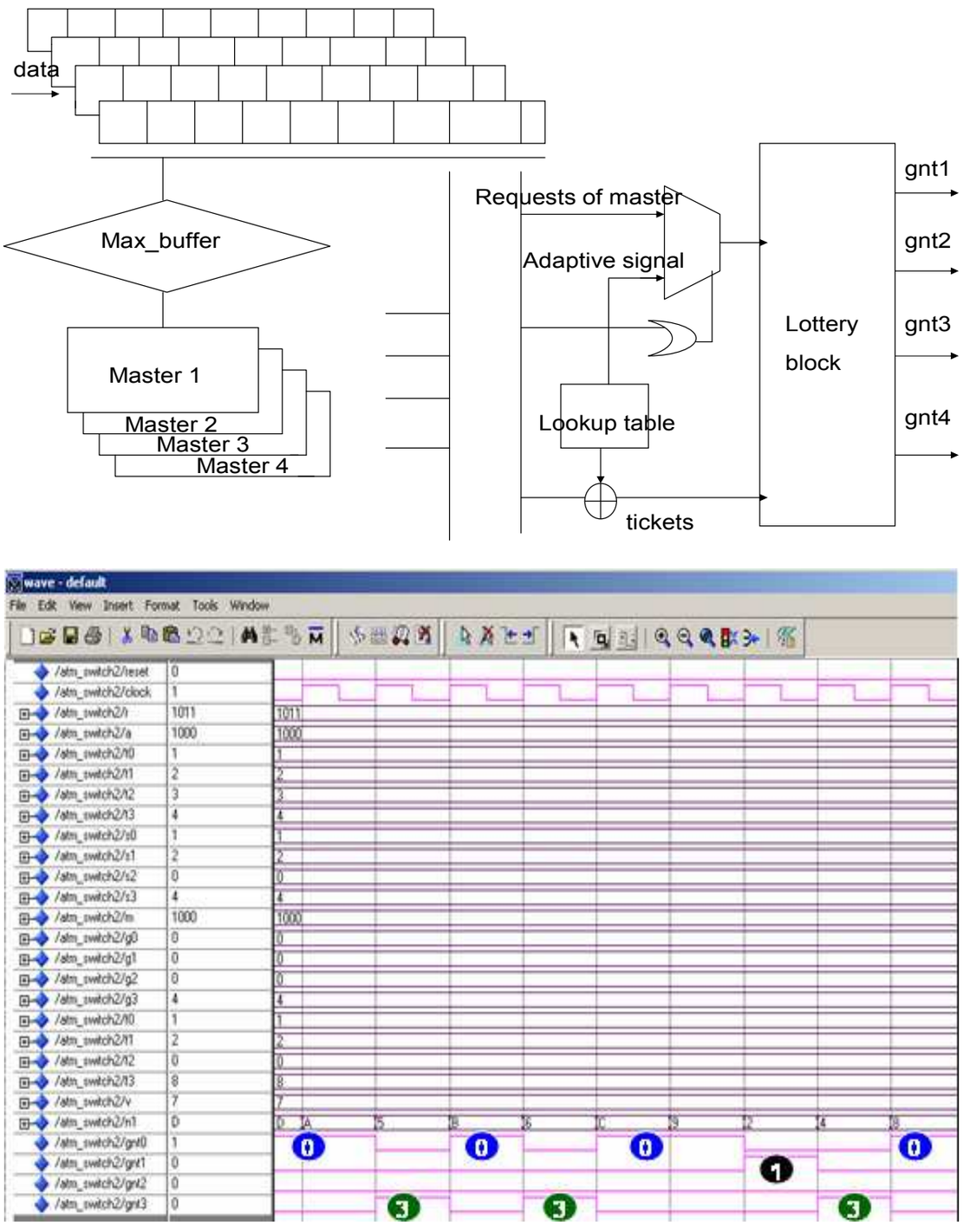


Fig 6. Structure & Simulation Results of ATM Switch arbiter

nonmembers of the class or category represented by the crisp set. Fuzzy sets boundaries are vague, and the transition from member to nonmember appears gradual rather than abrupt. A fuzzy set can be defined mathematically by assigning to each possible individual in the universe of discourse a value representing its grade of membership in the fuzzy set. This

grade corresponds to the degree to which that individual is similar or compatible with the concept represented by the fuzzy set.
The fuzzy arbiters are modeled using appropriate membership function and rules in such a way as to maximize the acceptance probability of the processors and distribute it evenly. In such systems, arbiters are used to resolve conflicts between processor requests shared bus. Typically, these conflicts are resolved by using two-stage arbitration schemes that employ policies such as random choice, daisy chaining, round-robin, etc. A new way of implementing these arbiters is the use of fuzzy logic to resolve resource request conflicts based on the system state and performance variables.

2.6.1 Working principal

The entire membership function can be divided into three segments: 0,1 and 2 as shown in Figure 7. The Y-axis shows the degree of membership (μ) as a value between 0 and 1.The X-axis shows the universe of discourse and is divided into three segments. Figure 3.8 shows how triangular input membership functions are formed in the fuzzification process. The calculation of the degree of membership (μ)can be categorized into three different segments: (a) In segment0: μ = 0, (b) In segment1: slope is upward from left to right, therefore: μ = (Input value – point 1) * slope1, μ is limited to max value of 1, (c) In segment 2: slope is downward from left to right, therefore: μ = 1 - (Input value –point 2) * slope 2 where μ is limited to a minimum value of 0.

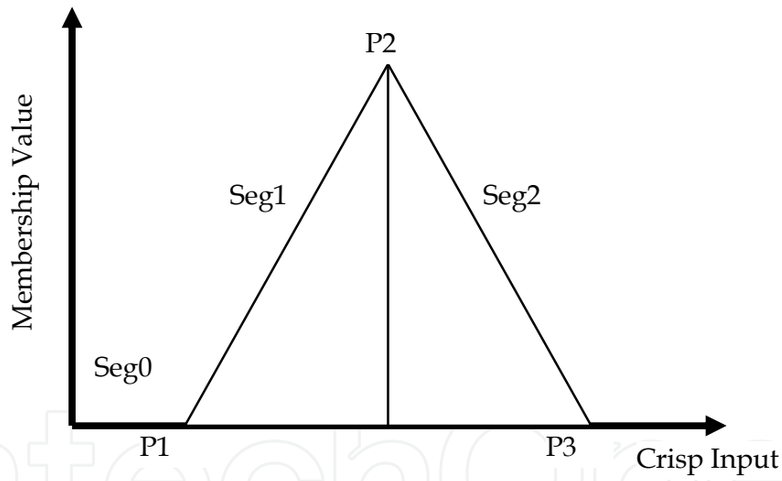


Fig. 7. Membership Functions

2.6.2 Design of fuzzy logic arbiter

Specifications: The arbiter has been designed for following specification.

- To improve Acceptance rate of each processor
- Using two level of arbitration first rule based and second priority based
- Designed for Three Masters.

Acceptance Rate calculation

Acceptance rate for each processor can be calculated as the ratio of master request granted with the master requested. If AARi is acceptation rate for ith processor, Pi.accept is number of request granted by FLA and Pi.nreq is total master request to FLA. Then Acceptance rate can be calculated as follows.

$$AARi = \frac{Pi.accept}{Pi.nreq} X100$$

For implementation, it will have two inputs as request and grant, output as Acceptance rate witch is 8-bit crisp value.

Fuzzification of inputs

In this step the degree of input is being determined by appropriate fuzzy sets via membership functions.

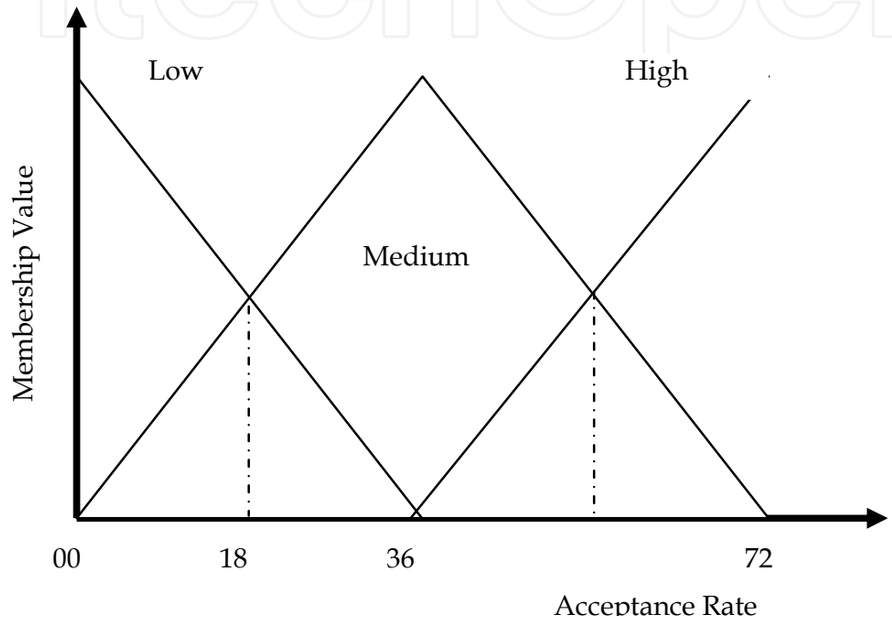


Fig. 8. Membership Functions for each processor

A membership function is a curve that defines how each point in the input space is mapped to a membership value (or degree). The inputs in this case are chosen to be the current acceptance rate of each processor. The input is a crisp numerical value limited to the universe of discourse which is in this case. Three membership functions are defined for each input; low, medium, and high, see in figure 8.

Rule Base Design

Once the inputs have been fuzzified, we know the degree to which each part of the antecedent has been satisfied for each rule. A set of rules have been defined for a fuzzy arbiter. The listing of the rules for a thee-input system is given bellow, where AP1, AP2 & AP2 are the current acceptance rates of input processor1 to processor3 respectively. The output is the processor selected (I1, I2 & I3). The rules have been chosen in such a way as to increase the acceptance rate of all processors, by selecting the lowest acceptance rate processor. In case of conflict i.e two processors acceptance rate are having in same category then problem will be solve by priority method. In such case processor 1 has highest priority and master 3 has lowest priority. Eg. In the table 1 fuzzy rule no 3, processor 1 and processor 2 has acceptance rate in low category but then processor 1 is selected due to high priority. Figure 12 shows block diagram of the rulebase module. Table 1 gives rule list.

Fuzzy rule	Ap1	Ap2	Ap3	Processor Selected
1	Low	Low	Low	I1
2	Low	Low	Medium	I1
3	Low	Low	High	I1
4	Low	Medium	Low	I1
5	Low	Medium	Medium	I1
6	Low	Medium	High	I1
7	Low	High	Low	I1
8	Low	High	Medium	I1
9	Low	High	High	I1
10	Medium	Low	Low	I2
11	Medium	Low	Medium	I2
12	Medium	Low	High	I2
13	Medium	Medium	Low	I3
14	Medium	Medium	Medium	I1
15	Medium	Medium	High	I1
16	Medium	High	Low	I3
17	Medium	High	Medium	I1
18	Medium	High	High	I1
19	High	Low	Low	I2
20	High	Low	Medium	I2
21	High	Low	High	I2
22	High	Medium	Low	I3
23	High	Medium	Medium	I2
24	High	Medium	High	I2
25	High	High	Low	I3
26	High	High	Medium	I3
27	High	High	High	I1

Table 1. Fuzzy Rule set for three processors

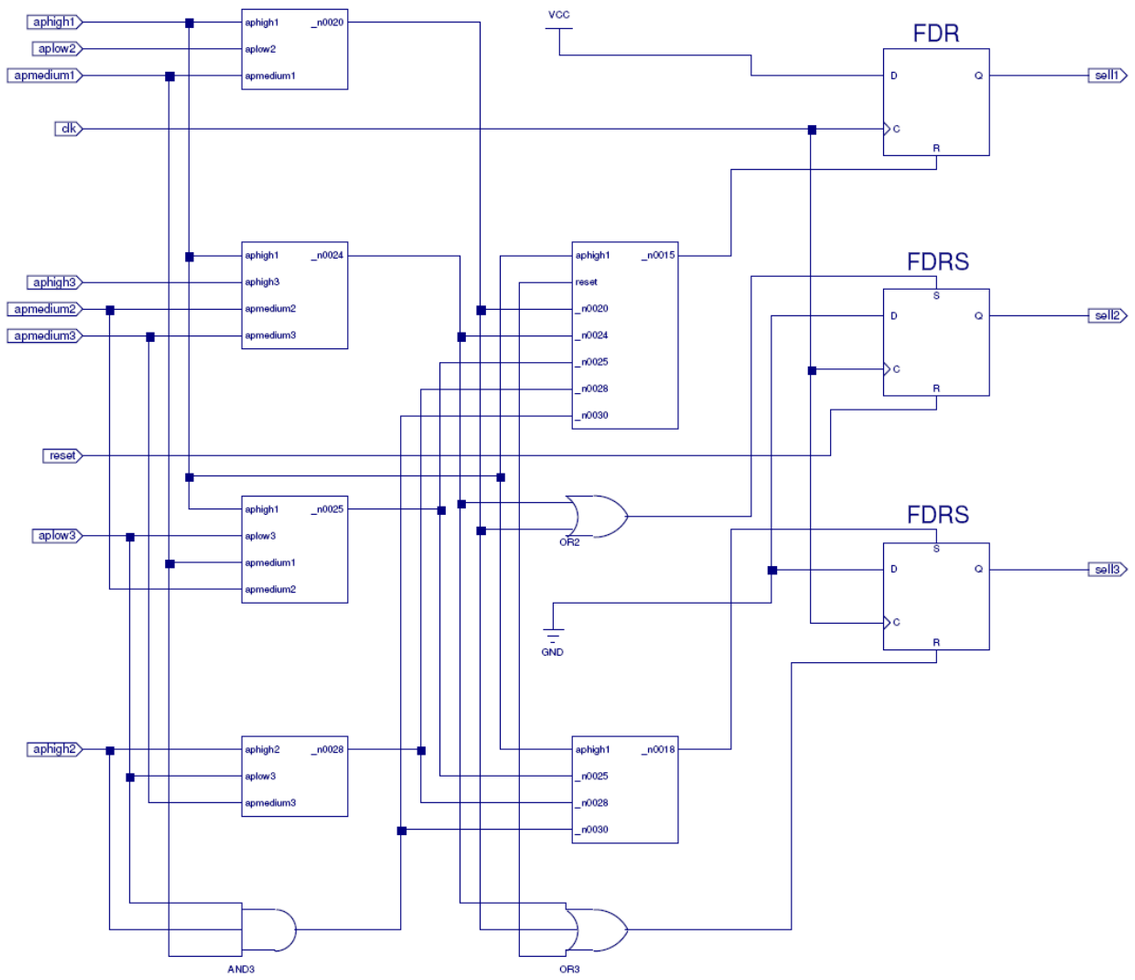


Fig. 9. Structure of Rule based Module

Fuzzy Arbiter for Three Processor

The figure 10 shows total structure for fuzzy logic arbiter for three processors. This structure has input as request signal Ireq1, Ireq2 & Ireq3 from respective three masters and output as grant signal I1grant, I2grant & I3grant to the masters. The structure is divided into three basic part from input side acceptance rate calculation, Middle part as fuzzification and output side as rule based module.

Fuzzy logic arbiter is complex to implement. The complexity increases exponentially with the increase in the number of processor. As the arbiter requires so many calculations to do, it will be slow for responding. Also it will hard to implement in FPGA because architecture consist of so many number of byte multiplier and divider that took huge hardware to implement.

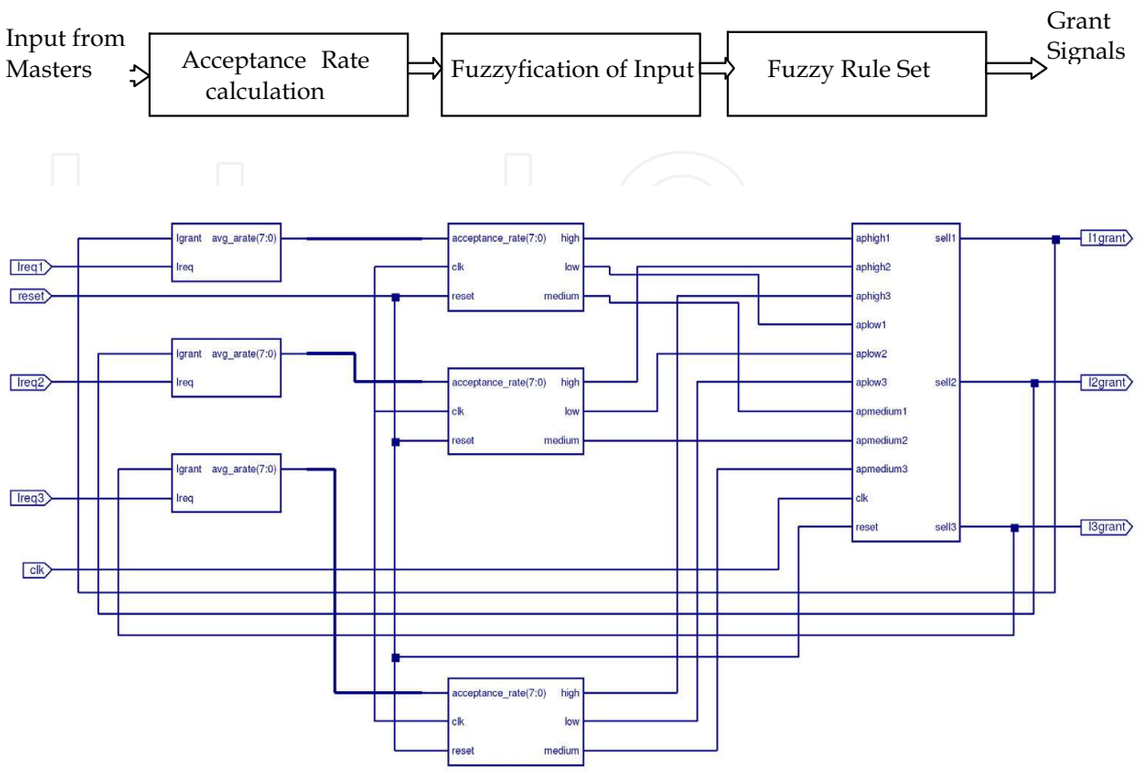


Fig. 10. Structure for fuzzy logic arbiter

3. Performance comparison of arbiters

Performance of the designed arbitration schemes has compared based on the parameter like Latency, Acceptance rate of Masters, Average Waiting time of masters & Shared bus bandwidth utilization by individual masters.

Average Latency (Cycles/word): It is a time delay between the moment something is initiated, and the moment one of its effects begins or becomes detectable. Ideally this should be zero of as minimum as possible.

Acceptance Rate: Acceptance rate is defined as percentage of how many times masters request for shared bus among how many times it request is granted and bus is allotted. Theoretically acceptance rate of every processor should be as high as possible.

Average Waiting Time: It is the average time for particular master in between request and grant of the shared bus. Average waiting time for every processor should be as low as possible.

Average Bandwidth Utilization: It is measure of shared utilized by different masters. The bus should be ideally equally utilized by all masters.

Figure 11 indicate that the ATM switch based arbiter gives optimum performance based on the monitoring parameters.

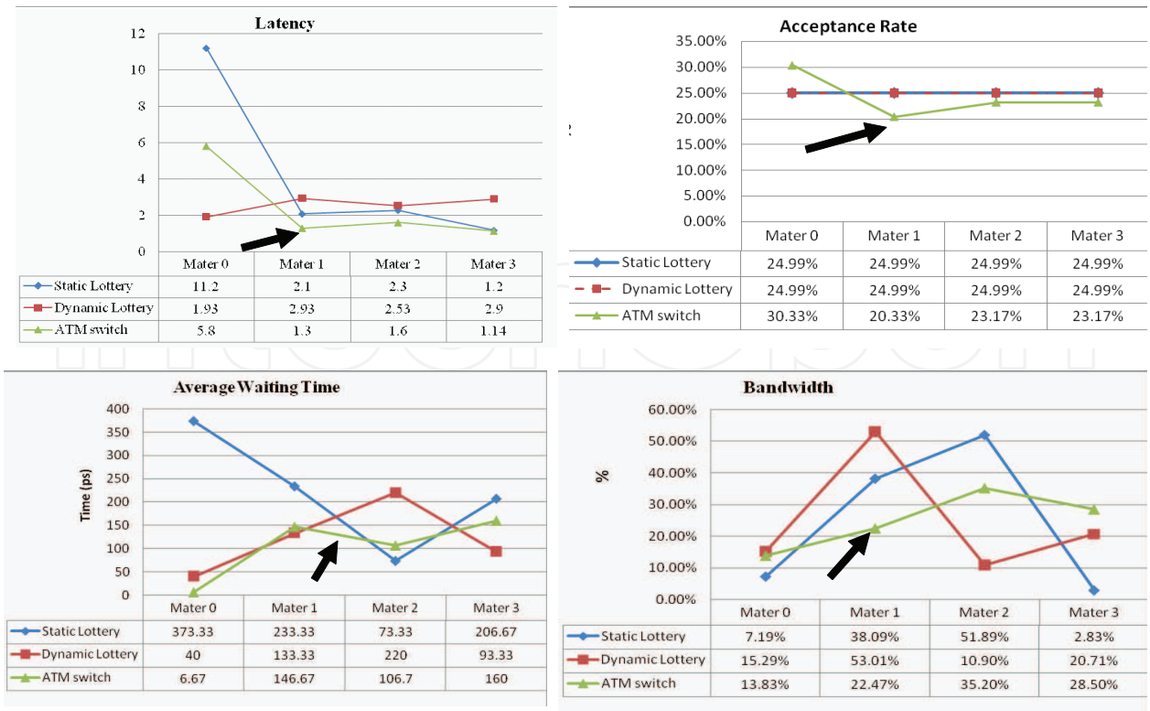


Fig. 11. Performance comparison of arbitration Techniques.

4. References

A.Y.Deshmukh, Dr.P.R.Bajaj, Dr.A.G.Keskar (2006). "Hardware implémentation of fuzzy logic controllers-Approach & Constraints", *KES 2006, Brighton 2006.*

Chang Hee Pyoun, et. al. (2003).“The Efficient Bus Arbitration Scheme In SoC Environment”, *IEEE International Workshop on System-on-chip*.

Dent,D.J. (2000). “System –on-Chip research leads to hardware/ software co-design degree”, *Frontiers in Educationa Conference, 2000. FIE 2000. 30th Annual Volume:2.*

G. Klir and T. Folger(1988). *Fuzzy Sets Uncertainty and Information*. New York: Prentice-Hall,

H. Diab,(2004). “Fuzzy logic arbiter for multiple-bus multiprocessor systems”, *IEEE Trans. On Systems, Man, And Cybernetics,*

Jun Xiao et. al.(2004). “Fuzzy controller for wall climbing microrobots”, *IEEE transactions on fuzzy systems, vol 12, no.4,*

K. Lahiri, A. Raghunathan (2006). “The Lotterybus on-chip communication architecture”, *IEEE Trans. On VLSI system.*

K. Lahiri, A. Raghunathan, and S. Dey (2000) “Performance Analysis of Systems with Multi-Channel Communication Architectures”, *International Conference on VLSI design, pp,530-537*

K. Lahiri, A. Raghunathan, and S. Dey, (2001)“System Level Performance analysis for designing on-chip communication architecture”, *IEEE Trans. Computer- Aided Des. Integr. Circuits Syst. Vol. 20.*

K. Lahiri, A. Raghunathan, G, Lakshminaray,(2001) “Lotterybus : A new high-performance communication architecture for system-on-chip designs” *Proc. Design Automation Conf. pp 15-20*

K. Lahiri, A. Raghunathan, G, Lakshminaray,(2001) "LOTTERYBUS : A new high-performance commun- ication architecture for system-on-chip designs" *Proc. Design Automation Conf. pp 15-20*

K.A Kettler, et.al (1995) "Modeling Bus Scheduling Policies for Real Time Systems", *16th IEEE Real Time Systems Symposium.*

Pawel Gepner et.al, (2006) "Multi-Core Processors: New Way to Achieve High System Performance", *IEEE Parallel Computing in Electrical Engineering (PARELEC'06)*

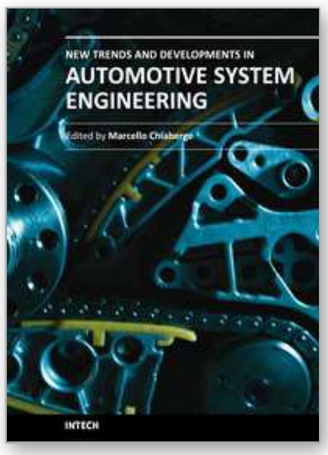
Sameep Singh, Kuldip Rattan (2003), "Implementation of a Fuzzy Logic Controller on an FPGA using VHDL", *IEEE*

Thomas D. Richardson, et. Al (2006) "A Hybrid SoC Interconnect with Dynamic TDMA-Based Transaction-Less Buses and On-Chip Networks", *IEEE 19th International Conference on VLSI Design (VLSID'06)*

Yi Xu , Li Li, Ming-lun Gao, Bing Zhang, Zhao-yu Jiang, Gaoming Du, Wei Zhang (2006)"An Adaptive Dynamic Arbiter for Multi- Processor SoC" *2006 IEEE*

Youngwoo Kim et.al (2003), "AMBA Based Multiprocessor System", *IEEE International Symposium on System-on-Chip.*

IntechOpen



New Trends and Developments in Automotive System Engineering
Edited by Prof. Marcello Chiaberge

ISBN 978-953-307-517-4
Hard cover, 664 pages
Publisher InTech
Published online 08, January, 2011
Published in print edition January, 2011

In the last few years the automobile design process is required to become more responsible and responsibly related to environmental needs. Basing the automotive design not only on the appearance, the visual appearance of the vehicle needs to be thought together and deeply integrated with the â€œpowerâ€ developed by the engine. The purpose of this book is to try to present the new technologies development scenario, and not to give any indication about the direction that should be given to the research in this complex and multi-disciplinary challenging field.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Preeti Bajaj and Dinesh Padole (2011). Arbitration Schemes for Multiprocessor Shared Bus, New Trends and Developments in Automotive System Engineering, Prof. Marcello Chiaberge (Ed.), ISBN: 978-953-307-517-4, InTech, Available from: http://www.intechopen.com/books/new-trends-and-developments-in-automotive-system-engineering/arbitration-schemes-for-multiprocessor-shared-bus

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2011 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen