International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 8, No.3 (May-2019)
http://dx.doi.org/10.12785/ijcds/080307

Fault-Tolerant Buffer Aware Round Robin Arbiter Design for NoC Architectures

Afshan Amin Khan[1], Roohie Naaz Mir[2] and Najeeb-ud-din[3]

[1]Department of Computer Science and Engineering, National Institute of Technology, Srinagar, India
[2]Department of Computer Science and Engineering, National Institute of Technology, Srinagar, India
[3]Department of Electronics and Communication Engineering, National Institute of Technology, Srinagar, India

Received 1 Feb. 2019, Revised 9 Mar. 2019, Accepted 9 Apr. 2019, Published 1 May 2019

Abstract: An arbiter is identified as one of the critical components of the NoC router. Among various arbitration schemes, Round-Robin arbiter is one among the popular arbitration schemes. In this work we have proposed an arbitration scheme that will be able to solve the problem of constant wait time of a conventional Round-Robin arbiter and will provided some additional features as well. The superiority of the proposed design is its ability to overcome this constant wait time, by identifying the real-time requirements of each port, based on information from respective buffers. The additional feature of the proposed algorithm is its fault-tolerant behavior for the errors related to buffer information. The proposed design is implemented using Vivado IDE and is verified on Zed-board Zynq-7000 FPGA platform. Simulation results reveal that the proposed algorithm has completely eradicated the drawback of constant wait time by performing arbitration dynamically. More importantly, it was observed that the proposed algorithm is tolerant to any temporary or permanent fault for deciding priorities. These major improvements in a Conventional round robin arbiter are achieved at the cost of 36% increase in area and a bonus 8% and 2% improvement in delay and operating frequency respectively.

Keywords: Fault Tolerant, Round Robin arbitration, Starvation Free, Buffer Aware, MPSoC, SoC, High Throughput

1. INTRODUCTION

The constant trend of decrease in the transistor feature size has opened new doors of research and analysis for the modern-day technocrats, such that more research efforts are intended towards the development of innovative and effective methods to overcome the unwanted effects arising due to decrease in transistor feature size. More importantly as the possibility of increasing the density of devices on the die has become higher, new avenues of developing efficient system architectures such as System on Chip (SOC) and Multiprocessor System on Chip (MPSOC) based systems have come to the picture. Modern day electronic system design has been revolutionized by the introduction of MPSoC paradigm [1, 2].Observing the growing number of cores on a single die, where these cores are expected to generate a large set of communication data, there is an immense need to connect them with specialized communication hardware. Realizing the needs of these systems, a feasible, high performance, reusable and scalable Thus Network on Chip (NOC) can be a promising solution [3]. The use of a communication network implemented together on a single die alongside the processing core can be a sophisticated and natural solution [4]. Thus, in near future, we can expect that MPSOCs or SOCs will be designed with integrated network on chip architectures. One such expected architecture is shown in figure 1. It depicts a 2D 8X8 Mesh topology, where each core tile is capable enough to process data and later forward this processed data to the desired location using the dedicated communication module associated with the core. Many such dedicated communication modules will form a network housed on to the chip, known as Network-on-Chip. A router, seemingly small but in itself is a complete system having various sub-modules. It is implied that now the speed, area and power of the network will depend upon the router as a basic unit [2].Thus, efficient implementation of internal modules of a router is of prime importance for achieving a high-performance router and ultimately a NoC fabric. An efficient arbitration unit is key to the performance of a high-speed switch or a router [5].The use of an effective arbitration scheme can affect the network performance [4]. Thus an efficient and reliable arbitration between the sources and destination becomes a major design concern. A significant problem in an arbiter design is its low-cost implementation, where cost relates to the amount of delay, area and other performance parameters as well[6].Typically, an ideal arbiter is expected to exhibit fairness, mutually

Email: afshan_58phd15@nitsri.net, naaz310@nitsri.net, najeeb@nitsri.net

exclusive property and first in first out policy of data transmission [7].

However, a more general and explanatory approach to devise a feasible arbitration algorithm can be summed in following points:

- An arbitration unit must be scalable, such that it can meet the varied architectural and functional needs.
- An arbitration unit should have parallel architecture to arbitrate between the requests in least possible time.
- An Arbitration unit should be a low-cost implementation in terms of area, as while developing arbitration strategies we are already on an area trade-off relation and consuming any more area can lead to degraded results.
- An Arbitration unit must have a decentralized approach for processing each request.
- An Arbitration unit must maintain the simplicity of the design such that it can easily be implemented using current VLSI implementation techniques [6].

The designs of arbiters that meet above or some major among the above mentioned criteria have been proposed in various literatures from time to time. For an arbitration unit housed on a chip or residing outside the chip, one of the most popular scheme is Round-Robin Arbitration scheme [6][23]. One method to implement a Round Robin Arbitration scheme is to develop a programmable priority encoder (PPE) and a state storing system to keep track of the priority pointer [8].A round robin arbitration scheme is relatively fair and starvation free. But for a Round Robin based arbiter it may happen that after n number of arbitration cycles, problems of larger waiting time and starvation among the low priority requestors comes into picture. Such problem if addressed with appropriate methodology can prove very beneficial for actual implementation and application of arbitration unit using VLSI tools. And will likely make such modified version of Round Robin algorithm as the prime choice of designers for use in NoC router and other places of application of an arbitration unit such as a crossbar switch or a bus-based architecture. Thus, in this work, we have concentrated over the design of an arbitration unit of a NoC router. By this approach, we expect that if we can devise efficient basic building blocks of a router the overall router performance will indirectly be improved as well. As a result, we can harness more benefits once such efficient routers are replicated and then connected to form a network of routers. Thus, we have focused on a Round Robin arbitration (RRA) unit for NoC router architecture wherein we are expecting to use its ability to switch between the requestors in a cyclic manner and also enable it to operate in Fault-Tolerant Buffer Aware mode so as to avoid any kind of starvation of certain ports, which is a considerable drawback of a conventional RRA. Since for modern day applications such as that of Internet of Things (IoT) based systems, the traffic patterns are not definite and can vary from uniform packet rates to hot-spot packet rates. Under such conditions, it’s desirable to implement an arbitration scheme that can distinguish between the low density and high density traffic patterns on a particular line and accordingly assign priority to inputs for gaining access to the resource.

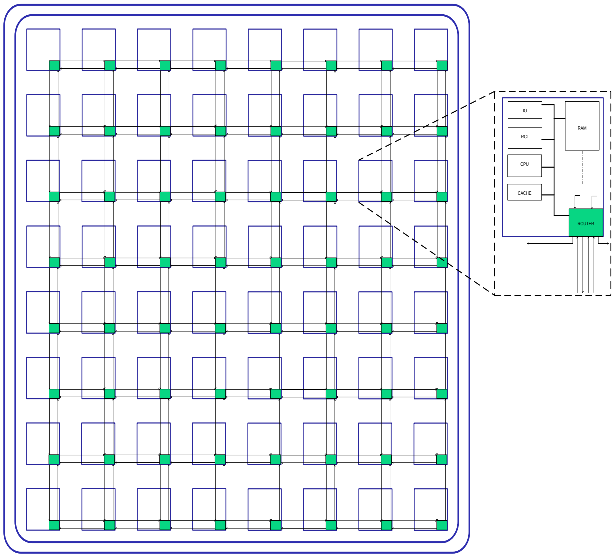


Figure 1. 2D 8x8 Mesh topology.

2. RELATED WORK

A wide variety of arbiters have been proposed in the literature keeping in view the architecture of NoC. We have made an effort to collect few relevant once here. Reference [5] presents two main designs which they named as Parallel Round-Robin arbiter (PRRA) and another as Improved PRRA. A PRRA is based on a simple binary search algorithm and even the IPRRA also maintains the simplicity of a binary tree. For design and necessary analysis, Synopsys Design tool has been employed, TSMC 180nm standard library has been used to synthesize the logical circuitry. Further, the designers have suggested that the combination of the two proposed algorithms can result in a version of algorithm that can be set for optimal values of gate delay, circuit delay and design complexity. As a conclusion, their designs are expected to achieve Round Robin fairness for all input patterns. Reference [9] provides a basic structure of a priority encoder based RRA that can make the decision between four request lines coming from four different devices. It devises a switch arbiter using RRA generator tool that can handle any number of masters. A detailed comparison between the proposed arbiter design, Ping Pong arbiter and a Programmable priority arbiter is provided in the work. As a conclusion, they declare that the proposed design shows a speed up of 1.9 and 1.24 times in comparison to Ping Pong arbiter and a programmable priority arbiter respectively. Reference [10] presents a novel arbiter and has named it as a credit controlled static priority (CCSP) based arbitration scheme. The main focus of this work is to eradicate the coupling of the rate of inputs and allocation granularity with latency. They have achieved the said objective by employing a CCSP scheme. A formal model is presented that describes how interactions between the requestors and the resources can be modeled using service curves. Authors have used a System C simulator platform employing a use case that uses H.264 video decoder and the proposed design is used as a DDR2 memory controller. From the simulation results, they have concluded that a CCSP based arbiter is expected to work properly with no increase in latency. In [11] authors propose a switch arbiter that can be used for on-chip-interconnection networks. The proposed design is named as tree scheduler and is in actual sense a multiplexer tree based cross–bar switch implementation. They notify that a RRA has a centralized priority pointer mechanism which results in higher latency. Thus propose a decentralized approach of arbitration that performs arbitration at every node. This approach is expected to give better results in terms of implementation area and latency as well. Authors have utilized SoC for High Definition Television as a platform to test their proposed design. As a conclusion, they declare that the proposed design is area efficient and has better performance when compared to conventional methods. Reference [8] suggests a Tiny Tera scheduler that combines iSLIP uni-cast scheduling with mRRM multicast scheduling. The authors further notify that the resulting algorithm is almost similar to the ESLIP used in Cisco's 12000 GSR router. Authors claim that the main contribution of their work is a fast Round Robin algorithm. They have identified that the basic principle to design a fast arbiter is to focus on the design of a fast programmable priority encoder (PPE) which is the heart of a Round Robin Algorithm. Thus they have focused on said module and have developed an improved programmable priority encoder. For performing the necessary design and analysis of the designs Texas Instruments TSC 5000 libraries have been used while maintaining similar area and time constraints for all designs under test.

Reference [12] propose a scheduling algorithm and have named it as deadline-based scheduling algorithm. The main focus of their work is to address the permanent Fault Tolerant scheduling problem in a heterogeneous system. Authors have provided a precise model of the system, the tasks, and the faults. They have compared a Fault Tolerant scheduling algorithm [13] and heterogeneous earliest-finish time algorithm [14] with their proposed algorithm and have concluded that their algorithm is a practical solution for task scheduling with fault tolerant capabilities for heterogeneous systems. Reference [15] identifies integration of SoCs as a growing challenge for the designer industry. Their work aims to address the feasibility of interconnection network components using asynchronous methods. As a result, authors present the implementation of packet-switch by employing asynchronous design methodology. Comparison between the proposed asynchronous design and its counter synchronous design reveals that asynchronous design performs better under conditions where the network scale is increased, on the contrary, synchronous counterpart will perform better when packet size is increased. Reference [16] presents a scheme and has named it as Extended Deficit Round Robin algorithm. The basic architecture of the proposed design can be explained as a hop-by-hop credit based flow control mechanism with modified Deficit Round Robin algorithm. The work provides network, node and port models, so as to express the functionality of the test setup. Authors after analysis of the results reveal that the proposed design performs better and is not affected by underlying quantum values used for Deficit Round Robin algorithm and also remains unaffected by packet size variations. Reference [17] analyses the effects of quantum size on Deficit Round Robin algorithm. It specifies that absence of segmentation and reassembly services along with varying packet size can lead a Round Robin algorithm to unfair allocation decisions. The main focus of the work is comparison of a Round-Robin algorithm having segmentation and reassembly capability with a Deficit Round-Robin algorithm without segmentation and reassembly services. Multiple workstations residing on the same LAN that receives data from the remote server in the form of packets has been employed as a simulation platform.

Reference [18] focuses on improving the performance of NoC by employing the methodology of prioritization of packets. The proposed algorithm relies upon the fact that the core which generates a large amount of data with respect to a threshold value will occupy higher ranks in the priority list and will be provided access to necessary resource that they are demanding. HDL code has been used to model the proposed algorithm using Modelsim platform under real-time traffic patterns such as MPEG, VOPD, MWD and DSP. From the simulation results, authors conclude that the proposed scheme of packet prioritization performs better than traditional Round-Robin and CAIS method, especially under non-uniform traffic patterns. Reference [19] presents a dynamic arbiter design by employing dual Round-Robin unit as a basic unit. The arbiter uses one Round-Robin unit to service the high priorities only while the other one is dedicated to service low priorities only. The design is expected to address the problem of starvation but at the cost of area trade-off. Reference [20] covers various methods to identify faulty dies in design industry. It also illustrates that a large amount of energy and effort is spent on finding these faulty dies at the final stage before sending the dies to the market. Thus provides a strong motivation to make the designs Fault Tolerant so that even if some dies skip this testing they still provide reliable operation even if they have some internal faults. This architectural level modification of the design will help not only the device to operate but also can save a lot of time if it’s recommended that the dies may be tested only for certain faults and all other tests may be skipped. Reference [24] designs a RTL code based hybrid arbiter so as to improve the performance of the arbiter in terms of latency and starvation. Reference [25] proposes a modified Index - based Round Robin arbiter. By doing so they expect improve the arbiter on account of power consumption, area and latency. The authors have used a VHDl based code to generate the necessary components of the design using Xilinx 13.2.

3. Proposed Design

The IEEE 802.16 standard was developed to ensure QoS for variable class of traffic with diverse QoS requirements. Scheduling of the packets traveling as traffic stays as the critical concern to ensure QoS [21, 2]. Apart from QoS one more important performance parameter is the overall throughput and reliability of a system. A conventional RRA and its modified versions such as weighted Round-Robin algorithm are among most commonly used algorithms for scheduling [21]. However, they are expected to provide depleted performance under non-uniform traffic situations [2]. As IoT is finding its applications in almost every field, it is evident that the processors and evaluation systems will receive a varied amount and pattern of data streams depending upon the nature of IoT device connected to them. It is expected that the communication fabrics of these systems should be able to comprehend the varied needs of the traffic arriving at each input port. Thus, NoC being the communication fabric needs to be updated according to the growing needs of the application traffic. Since an arbiter plays a critical role for timely and reliable packet delivery from the source to the destination. As a result, we realized the need of an arbiter for NoC that is aware of the varied traffic rates arriving at its input ports and more importantly provide reliable operation even when some internal faults may occur is of prime importance. Thus, we have proposed an arbiter unit which is able to update the priority list dynamically based on the traffic pattern of each input port. We have adopted buffer capacity flag status of buffers as a mean to let the arbiter know about the different rates of traffic arriving at different directions. For a particular direction in case the traffic rate is high, then buffers will be full early in these directions as compared to those directions which have a low rate of arriving data, thus if a direction raises its buffer full (b_f) flag then it needs immediate attention. Furthermore, the peculiar feature of the proposed design is that it identifies any glitch in the priority assigning b_f signal. In our design we have related a glitch of b_f signal to either of the two viz. either a condition when a fault occurs at b_f signal due to coupling from neighboring signal lines resulting in logic 1 being forced on to a particular line of b_f bus or condition when a b_f signal line holds logic 1, due to stuck at 1 fault. We have concentrated on faulty cases arising due to enforcement of logic 1 only, since it leads the system to a critical deadlock condition. Faulty conditions arising due to logic 0 being coupled from neighboring lines or a stuck at 0 fault have not been considered in this work. As the system will not enter deadlock but will still operate as a general RRA due to enforcement of logic 0 at any b_f line.

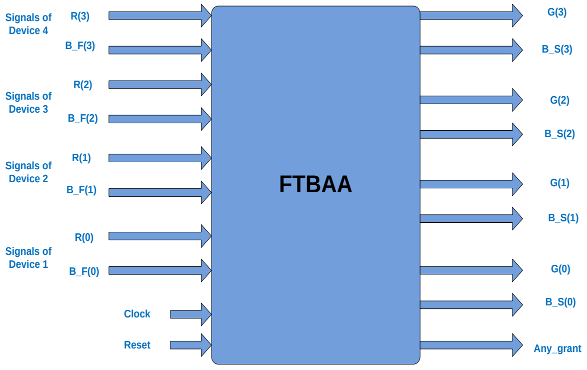


Figure 2. Functional block diagram representation of fault tolerant buffer aware arbiter.

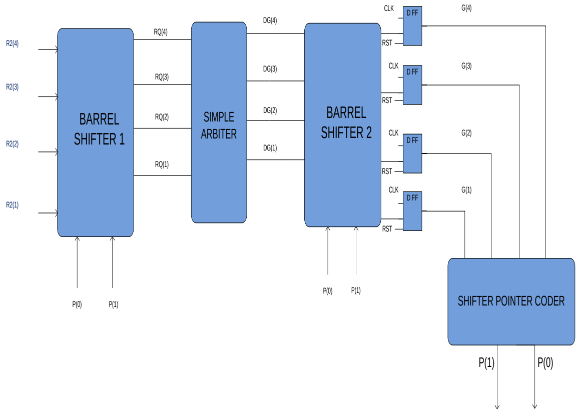


Figure 3. Internal block diagram of a RRA.

Buffer Aware arbiter presented in [2] can be regarded as a basic Buffer Aware Round-Robin arbiter (BARRA) for NoC. Whereas the design we have proposed in this work can be regarded as a Fault-Tolerant Buffer Aware arbiter (FTBAA) for NoC. As the nature of the BARRA is such that if all the request signals are high i.e. each of them is asking for a grant to access the common resource but b_f signal of a particular direction is high for infinite amount of time then in that case the arbiter will end up granting the access to this particular requestor for infinite amount of time, even if request signals of all four directions are high. This hold condition can be either intentional or else a fault in the respective b_f line. The situation can become even worse if all requests are high other than the one whose b_f signal has got stuck viz. A glitch condition like b_f=1000 while r=0111. Then under such conditions, BARRA will not provide any grants and will continue to waste arbitration cycles as long the current situation prevails. By this, we realized that a b_f signal has a total monopoly in deciding the final grant to access the shared resource. Therefore, identification and neutralization of any faulty condition at this control signal is crucial for the reliable operation of an arbiter unit. Our proposed design can be regarded as a hybrid arbiter design which is not only aware of its buffers but also is able to provide Fault-Tolerant operation with respect to priority deciding b_f signal lines. The control circuitry is modified in such a way that b_f is not the only signal that has a monopoly in deciding the priority. It's made sure that only a minimal circuitry is appended to a general RRA to develop an arbiter that is Buffer Aware and is also expected to work in Fault-Tolerant mode once any unwanted glitches may occur at priority deciding b_f signal.

Figure 2 represents the block diagram of the proposed algorithm. The basic component of an FTBAA is a fixed priority arbiter unit. This fixed priority arbiter unit is appended with certain other components to achieve a RR unit [19]. Figure 3 shows the internal block diagram of the RR Unit [19]. The FTBAA can be taken as a basic RR arbiter appended with low area additional circuitry for providing fault tolerant and buffer aware capabilities. Considering a faulty situation for a BARRA scheme where a b_f bit of particular requestor remains high due a stuck at 1 fault or due to any unwanted coupling between the internal signals, while its respective request signal has gone low and the next requestor is asking for a grant. BARRA will respond by providing no grants at all and renders the shared resource unassigned and the overall system as defunct, as can be observed from simulation result given in Figures 15 and Figure 16. Thus overall system enters into a deadlock mode until the fault exists in the system. Resulting in wastage of arbitration cycles, as BARRA ignores the fact that other requestors are still in need of the shared resource. Thus because of the occurrence of a glitch in the input signals the channel is not assigned to any of the requestors. If such a situation continues in the system for infinite amount of time, then the resources will remain idle for infinite amount of time. As a result, n number of arbitration cycles will be wasted, that will waste energy as well as decreases the overall throughput of the system. Any technique that will avoid this halt of the system will surely have a direct impact on the energy saving and is also expected to increase the throughput as well.

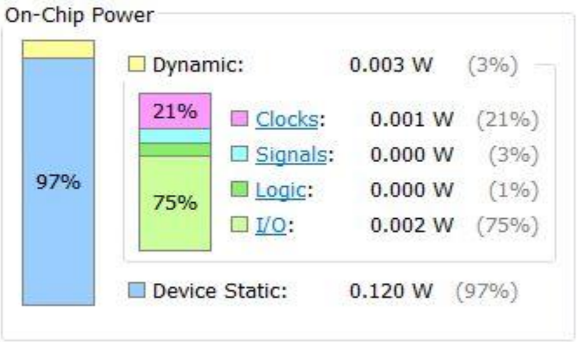
Considering the similar situation for an FTBAA scheme, here assigning of grants will mainly be controlled by the status of the respective request signals of each direction. Thus when there are any glitches at a particular b_f signal and the request signal of another requestor is high whose respective b_f signal is low, then in that case, this requestor will be assigned a grant even if, b_f signal of some other requestor is high but whose request signal is low. In FTBAA scheme we have been able to achieve arbitration in such a way that the arbiter by-passes the gated control mechanism once it identifies any glitch in the b_f signal and operates as a general RRA. This strategy does not allow the system to enter into the deadlock mode thus avoiding the wastage of arbitration cycles as well as the energy of the overall system. In order to achieve the said operation of a FTBAA we have combined b_f with their respective request signals so as to decide the final grant. An FTBAA has 10 input signal that arrive at the input side of the arbiter which include Reset, Clock, r(4), r(3), r(2), r(1), b_f(4), b_f(3), b_f(2) and b_f(1). Reset and Clock signals are according to the convention of digital design where as r(4)to r(1)are the request signals from four different requestors, a logic 1 on any of these signal lines means that respective requestor is demanding access to the resource. Similarly, b_f(4) to b_f(1) are the buffer flags of the respective requestors, a logic 1 at any of these means that respective requestor's buffer is full. In a general RRA r(4) to r(1) are checked every arbitration cycle and if any of them is logic high it is stacked and grants are assigned on the basis of their position in the stack on a cyclic basis. Although RRA is an efficient way of arbitration however is likely to face starvation once the traffic patterns become non-uniform [2]. A modified RRA can be expected to become a preferred choice once the proposed Fault-Tolerant Buffer Aware mechanism can be appended to a conventional version of RRA. In order to achieve this, we have divided the circuit into following main modules: A Grant regulation logic that allows the arbiter to operate either as an FTBAA or as a conventional RRA, A Control generation circuit that sends necessary control signals for the multiplexer array inside grant regulation unit and finally the conventional Round Robin circuit.

4. RESULTS AND DISCUSSION

In order to perform the necessary simulation and analysis of the designs, we have implementation the design using Vivado IDE followed by FPGA based verification on Zed-board Zynq-7000 series.HDL based modeling has been employed to design the respective logic circuits. For sake of simplicity in identification and analysis, we have classified the designs into following six categories and named them accordingly. Type I depicts a Matrix arbiter (M.A) [22], Type II depicts a Buffer Aware M.A [2], Type III depicts a Fault Tolerant M.A, Type IV depicts a conventional RRA [19], Type V depicts a Buffer aware RRA [2] and Type VI depicts an FTBAA.

Table I represents the performance parameters obtained after simulation of the designs in Vivado. Here number of LUT/FF represents the number of four input look up tables and Flip Flop used by the design for its implementation i.e. can be related to the area of the design. Delay represents the maximum data path delay of the design. Power represents the total power consumed by the design where total power is a sum of the static and dynamic power of the FPGA device. Since the major portion of the FPGA module is unused hence static power was found to be the dominant one among the two. Therefore, for better understanding of the dynamic power consumed by the design, illustrations are provided from figures 4 to 9 for each design type. Freq. represents the maximum operating frequency of the design. ORT represents the maximum output required time after the clock. All parameters have been evaluated using Xilinx

Vivado IDE design suit whereas frequency and ORT has been evaluated using Xilinx ISE design suit, so as to maintain the simplicity of the analysis. An HDL based test bench is used to test the functionality of the design.

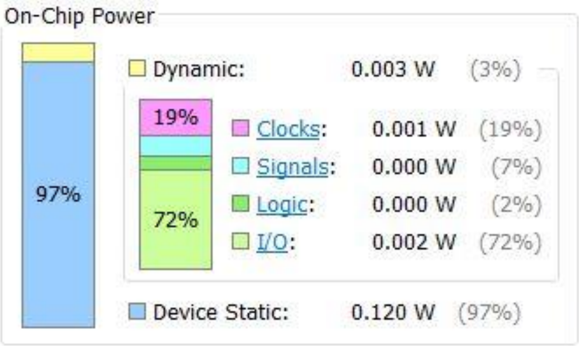

On-Chip Power

Dynamic: 0.003 W (3%)

Clocks: 0.001 W (21%)
Signals: 0.000 W (3%)
Logic: 0.000 W (1%)
I/O: 0.002 W (75%)

Device Static: 0.120 W (97%)

Figure 4. Power consumption for Type I.

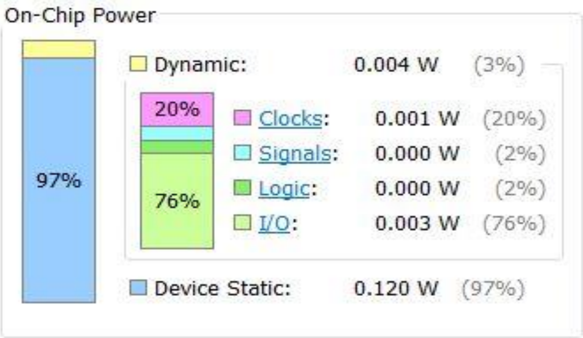

On-Chip Power

Dynamic: 0.003 W (3%)

Clocks: 0.001 W (19%)
Signals: 0.000 W (7%)
Logic: 0.000 W (2%)
I/O: 0.002 W (72%)

Device Static: 0.120 W (97%)

Figure 5. Power consumption for Type II.

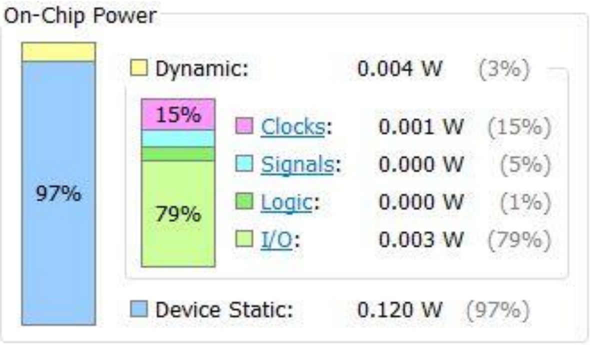

On-Chip Power

Dynamic: 0.004 W (3%)

Clocks: 0.001 W (20%)
Signals: 0.000 W (2%)
Logic: 0.000 W (2%)
I/O: 0.003 W (76%)

Device Static: 0.120 W (97%)

Figure 6. Power consumption for Type III.

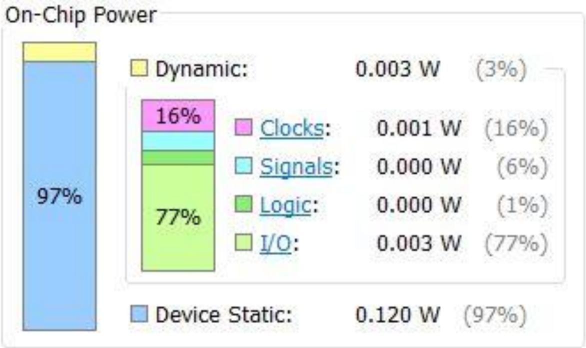

On-Chip Power

Dynamic: 0.004 W (3%)

Clocks: 0.001 W (15%)
Signals: 0.000 W (5%)
Logic: 0.000 W (1%)
I/O: 0.003 W (79%)

Device Static: 0.120 W (97%)

Figure 7. Power consumption for Type IV.

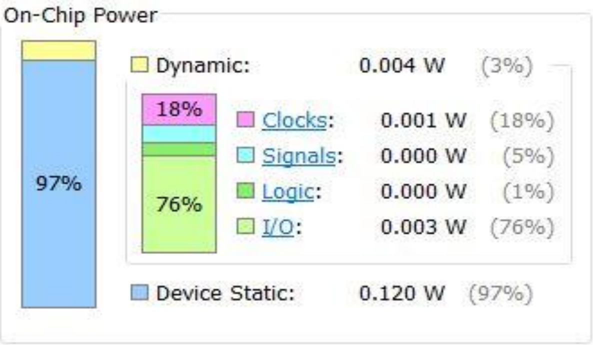

On-Chip Power

Dynamic: 0.003 W (3%)

Clocks: 0.001 W (16%)
Signals: 0.000 W (6%)
Logic: 0.000 W (1%)
I/O: 0.003 W (77%)

Device Static: 0.120 W (97%)

Figure 8. Power consumption for Type V.


On-Chip Power

Dynamic: 0.004 W (3%)

Clocks: 0.001 W (18%)
Signals: 0.000 W (5%)
Logic: 0.000 W (1%)
I/O: 0.003 W (76%)

Device Static: 0.120 W (97%)

Figure 9. Power consumption for Type VI.

TABLE I. PERFORMANCE PARAMETER CALCULATION TABLE

Design Type	**Performance Parameters**				
	LUT /FF	***Delay (ns)***	***Power (mw)***	***Freq. (MHz)***	***ORT (ns)***
Type I	15 /6	7.663	123.0	257.377	7.111
Type II	18 /6	7.758	123.0	250.733	7.540
Type III	19 /6	7.543	124.0	258.873	7.210
Type IV	7 /4	5.339	124.0	263.612	5.447
Type V	11 /4	5.339	123.0	263.612	5.447
Type VI	11 /4	4.916	124.0	268.280	5.447

The test bench is designed in such a way that it tests the design for all possible combination i.e. 2^n combinations, where n is the number of input variables. All these combinations are applied as input, resembling to a random traffic model. In our design r depicts the request status from a particular direction or requestor. Since we have assumed four directions thus r is a four-bit vector. Where each bit represents a requestor's request line. Logic 1 at this request line depicts that the requestor has some data to send to a particular channel whereas logic 0 at the request line depicts that the requestor has no data to be sent. Similarly, b_f is a four-bit vector and each bit belongs to a buffer flag of particular direction.

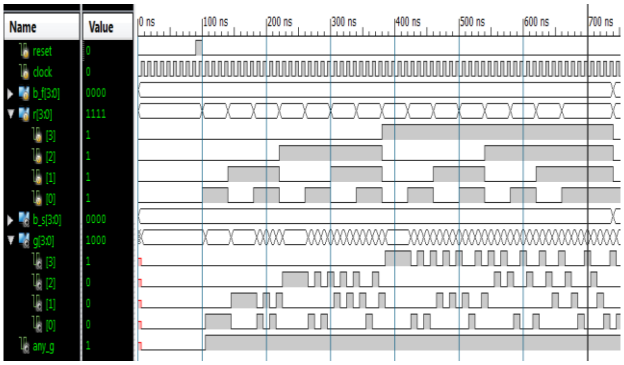


Figure 10. Response of FTBAA for b_f=0000.

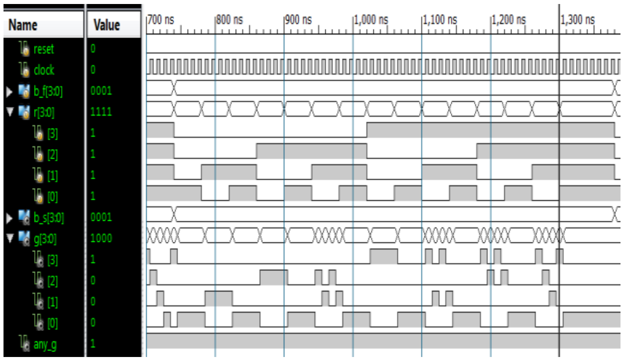


Figure 11. Response of FTBAA for b_f=0001.

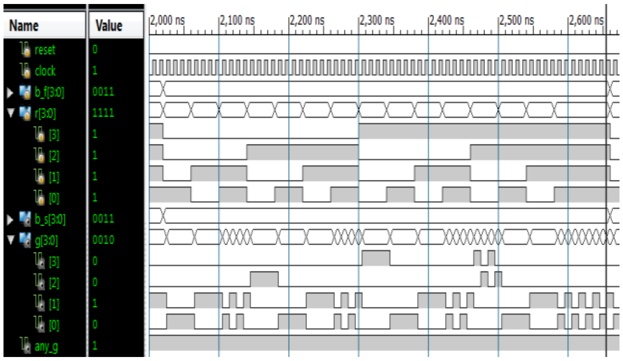


Figure 12. Response of FTBAA for b_f=0011.

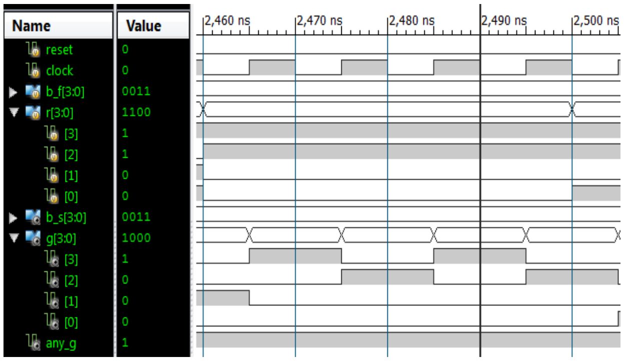


Figure 13. Response of FTBAA forb_f=0011while r=1100.

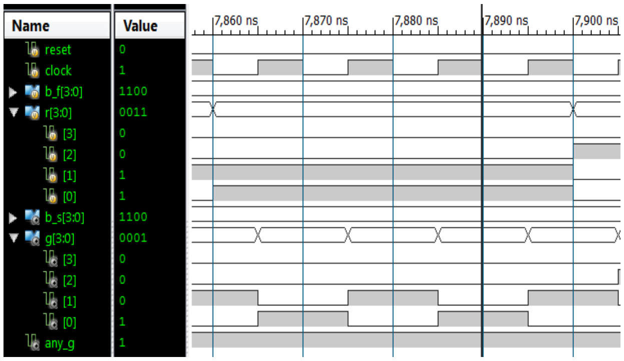


Figure 14. Response of FTBAA for b_f=1100 while r=0011.

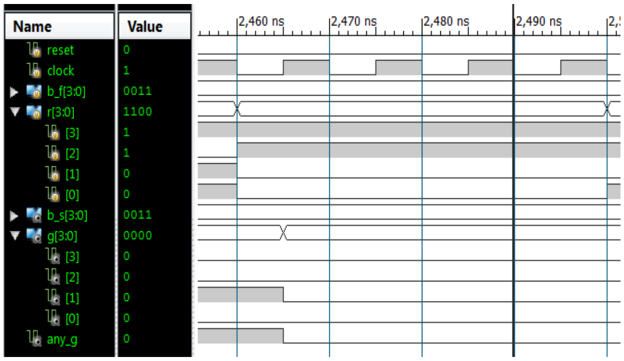
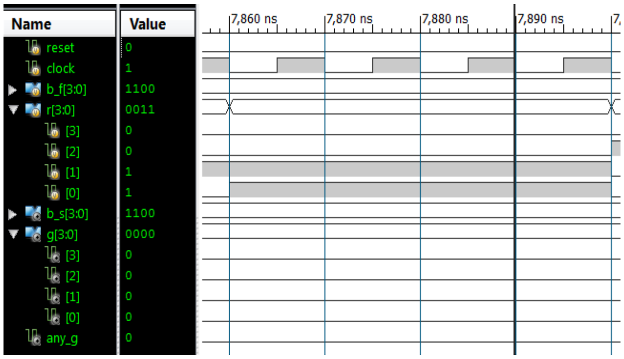


Figure 15. Response of BARRA for b_f=0011 while r=1100.

Logic 1 at b_f signal of a particular requestor depicts that its buffers are full where as logic 0 at b_f line depicts that the buffers of respective direction are not yet full and can bear some more data. The simulation results generate reliable output waveform for respective input combination. The output lines comprise of four grant signals depicted by four-bit vector g and also a common any_g signal to depict occurrence of a grant. Signals b_s is another output signal and its status ensure that the buffer flags are read properly in each arbitration cycle. A reset signal is included to reset the design to its initial state. As the reset is enabled all outputs and inputs are initialized and only when the reset is disabled again the design starts its normal operation from 100 ns as in figure 10.

Simulation results for the design given in figure 10 depicts the operation of the FTBAA under the condition when b_f=0000 and r varies from 0000 to 1111i.e. all 2^n combinations are applied as test input. Since b_f =0000 means that there is no load on buffers and request signal are fed in number sequence so as to test it for random request pattern, the grants under this condition will be provided on the basis of general Round-Robin mechanism. It’s worth observing that the grants are awarded on mutually exclusive basis i.e. none of requestor’s grant period overlaps with any other grant period, which is one of the prime features of a feasible arbiter design.

Here an arbitration cycle is equivalent to one clock period, thus the minimum amount of time a requestor is awarded a grant is equivalent to one complete clock period and only then grant is assigned to others in the list. However according to varying need of the traffic arriving at the input a particular request may gain the access to the shared resource for more than one arbitration cycle, which is one of the key features of the proposed design. Figure 11 depicts a condition where b_f=0001 and r varies from 0000 to 1111 i.e. this time also all 2^n combinations are applied as test input. In this condition as the buffer flag of requestor0 is raised depicting a condition of heavy load on its input thus demanding more service than others. FTBAA responds to such an immediate condition by blocking requests of all other requestors and dedicates arbitration cycles to this requestor as long as r(0) remains high. However for conditions where r(0) goes low but b_f(0) maintains a logic high condition. An FTBAA identifies this condition as a glitch case, thus analyzes the status of other request signals as well. In case some other requestors are asking for the grant then FTBAA distributes arbitration cycles between these requestors, while ignoring the stuck at 1 status of b_f(0). Point to note here is that this dynamic feature of varying the priority is not possible with a conventional RRA. Now if a condition arises where another requestor raises its buffer flag, as depicted in figure 12 then in that case all other requests will be blocked and grants will be shared between these two heavily loaded requestors only.



Name	Value
reset	0
clock	1
b_f[3:0]	1100
r[3:0]	0011
[3]	0
[2]	0
[1]	1
[0]	1
b_s[3:0]	1100
g[3:0]	0000
[3]	0
[2]	0
[1]	0
[0]	0
any_g	0

Figure 16. Response of BARRA for b_f=1100 while r=0011.

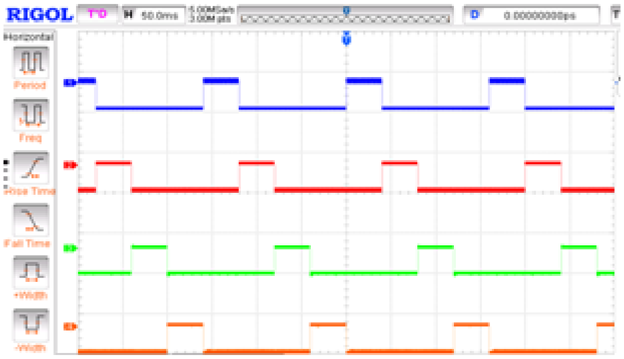


Figure 17. Response of FTBAA for r=1111.

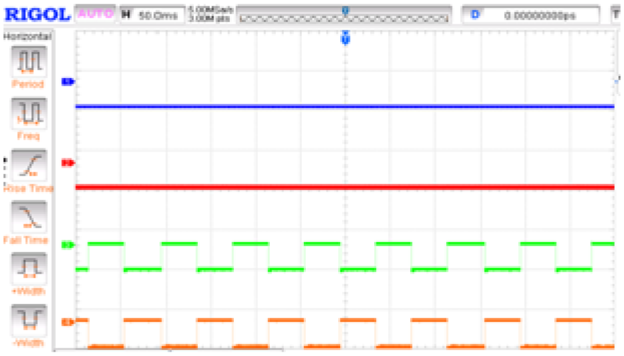


Figure 18. Response of FTBAA for b_f=1100 while r=0011.

Since there may occur conditions where b_f=0011 and r=1100 i.e. conditions where respective request line is low depicting no request from the requestor but due a false trigging at its respective b_f signal or occurrence of a stuck at 1 fault at its flag it seems as if the requestor is receiving high data. FTBAA is able to identify such faulty conditions and performs arbitration only among those requestors whose request signal r is high. Figure 13 and 14 respectively depict the response of an FTBAA for two such conditions where b_f=0011 while r=1100 and b_f=1100 while r=0011. In figure 13 it can be observed that from the first positive edge of the clock signal, grant signal g(3) attains a state of logic 1, representing grant for requestor 3, followed by g(2) going high representing grant for requestor 2. Such condition of round-robin grants between these two would continue as long as a valid high priority situation exists. Figure 20 plots the Power-Delay-Product (PDP) of each design type, as can be observed that apart from above-mentioned advantages in the nature of response to particular input conditions, we were more importantly able to achieve a lower value of PDP as compared to other algorithms.

As a part of verification, we physically dumped the design on a Zynq-7000 FPGA bearing part number xc7z020clg484-1. The test setup was arranged in such a way that the inputs were provided from the FPGA board and the output response of the design was observed on the oscilloscope, whose probes were connected to the output ports of the FPGA.

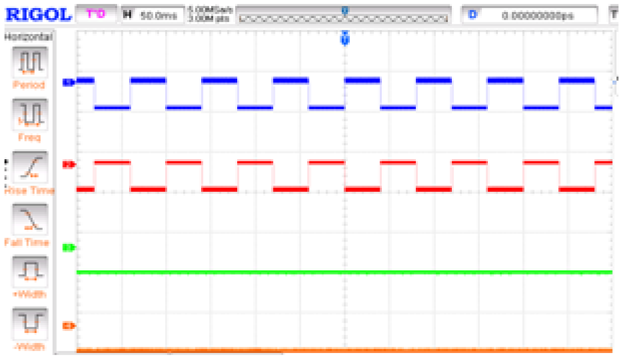


Figure 19. Response of FTBAA for b_f=0011 while r=1100.

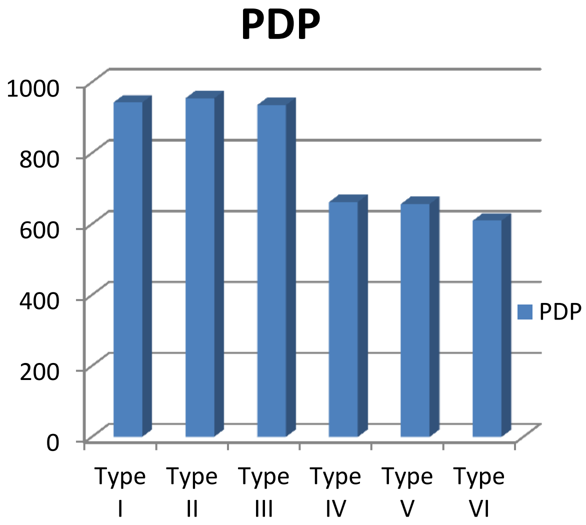


Figure 20. Power-Delay-Product comparison results.

Response from the oscilloscope was recorded and is presented here in figure 17, figure 18 and figure 19, for the input conditions for which already simulations results are depicted in figure 10, figure 13 and figure 14 respectively. This shows that the behaviour of the design, when dumped on an FPGA, is exactly the same as the simulated behaviour of the design implemented in Vivado IDE design suite. It was observed that the output response displayed on oscilloscope screen is as expected while maintaining the mutually exclusive nature of grants and their repetition after a fixed interval of time, since inputs are continuously demanding. The response of the design for other input conditions was tested as well and was found in accordance with simulation results.

5. Conclusion

Various arbitration schemes for NoC architecture were discussed and it was realized that there is a need of an arbiter that is able to maintain the simplicity of a Round-Robin algorithm and is able to deliver services according to the varying traffic needs of each competing port. In this work, we have presented a Fault-tolerant buffer aware algorithm for arbitration and have named it as FTBAA. An FTBAA is expected to reduce the constant wait time of a conventional RRA and also enable the fault tolerant mode of operation. We have implemented and analyzed the design using Vivado IDE and targeted the design for a Zynq-7000 series platform. Area, delay, power, frequency and output return time after clock are chosen as performance parameters. On analyzing the comparison results its observed that the proposed design operates with 4% increase in operating frequency, 36% decrease in delay, 27% decrease in area,1% increase in power and 23% decrease in output return time after clock in comparison to a Matrix arbiter. Whereas on comparing the proposed design with RRA arbiter it is observed that proposed arbiter operates with 2% increase in operating frequency,8% decrease in delay, 36% increase in area, with same power consumption and similar output return time after clock. More importantly the inclusion of the fault-tolerant mechanism is expected to prevent the wastage of arbitration cycles leading the system towards better throughput capabilities. For the sake of verification, we implement the design on a Zed-Board based FPGA for reliability check and comparison with simulation results was done using an oscilloscope. Hence, we concluded that with the advent of latest technologies it is evident that the modern-day systems will be surrounded by many equipments in the form of sensing devices, actuators and communication modules. Thus, reliable communication and higher throughput will remain as the primary concern. An arbiter being a critical block of a NoC router needs to be highly efficient and reliable as well. Therefore, proposed architecture is expected to improve the reliability and throughput of an arbiter block and a router as a whole.

Acknowledgment

This publication is an outcome of the R&D work undertaken project under the Visvesvaraya PhD Scheme of Ministry of Electronics & Information Technology, Government of India, being implemented by Digital India Corporation.

References

[1] W. Wolf, A.A. Jerraya, G. Martin Multiprocessor system-on-chip (mpsoc) technology,IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., 27 (10) (2008), pp. 1701–1713.

[2] A. A. Khan, R. N. Mir and Najeeb-ud-din, "Buffer aware arbiter design to achieve improved QoS for NoC," TENCON 2017 - 2017 IEEE Region 10 Conference, Penang, 2017, pp. 2494-2499.

[3] Jain, Kunj, et al. "Problems encountered in various arbitration techniques used in noc router: A survey." Electronic Design, Computer Networks & Automated Verification (EDCAV), 2015 International Conference on. IEEE, 2015.

[4] Samman, Faizal Arya. "Network-on-chip with an arbitration control for balancing throughputs in multiprocessor applications." Electrical Engineering and Informatics (MICEEI), 2014 Makassar International Conference on. IEEE, 2014.

[5] Zheng, Si Qing, and Mei Yang. "Algorithm-hardware codesign of fast parallel round-robin arbiters." IEEE transactions on parallel and distributed systems 18.1 (2007): 84-95.

[6] Jou, Jer Min, Yun-Lung Lee, and Sih-Sian Wu. "Model-driven design and generation of new multi-facet arbiters: from the design model to the hardware synthesis." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 30.8 (2011): 1184-1196.

[7] Oikonomou, Kostas N. "Ideal arbiters: Analysis and design." AT&T technical journal 66.2 (1987): 78-96.

[8] Gupta, Pankaj, and Nick McKeown. "Designing and implementing a fast crossbar scheduler." IEEE micro 1 (1999): 20-28.

[9] Shin, Eung S., Vincent J. Mooney III, and George F. Riley. "Round-robin arbiter design and generation." Proceedings of the 15th international symposium on System Synthesis. ACM, 2002.

[10] Akesson, Benny, et al. "Real-time scheduling using credit-controlled static-priority arbitration." Embedded and Real-Time Computing Systems and Applications, 2008. RTCSA'08. 14th IEEE International Conference on. IEEE, 2008.

[11] Lee, Kangmin, S-J. Lee, and H-J. Yoo. "A high-speed and lightweight on-chip crossbar switch scheduler for on-chip interconnection networks." Solid-State Circuits Conference, 2003. ESSCIRC'03. Proceedings of the 29th European. IEEE, 2003.

[12] Liu, Jing, et al. "Task Scheduling with Fault-Tolerance in Real-Time Heterogeneous Systems." Journal of Systems Architecture (2018).

[13] Benoit, Anne, Mourad Hakem, and Yves Robert. "Fault tolerant scheduling of precedence task graphs on heterogeneous platforms." Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on. IEEE, 2008.

[14] Topcuoglu, Haluk, Salim Hariri, and Min-you Wu. "Performance-effective and low-complexity task scheduling for heterogeneous computing." IEEE transactions on parallel and distributed systems13.3 (2002): 260-274.

[15] Xu, Jun, Reza Sotudeh, and Mark B. Josephs. "Asynchronous Packet-Switching for Networks-on-Chip." Application of Concurrency to System Design, 2006. ACSD 2006. Sixth International Conference on. IEEE, 2006.

[16] Lee, Teck Peow, and John Siliquini. "Deficit Round Robin with hop-by-hop credit based flow control." TENCON 2007-2007 IEEE Region 10 Conference. IEEE, 2007.

[17] Lee, Teck Peow, Tarith Devadason, and John Siliquini. "The effects of quantum size on Deficit Round Robin performance." TENCON 2007-2007 IEEE Region 10 Conference. IEEE, 2007.

[18] Mohandesi, E., and M. Mohandesi. "Improving performance of NoCs by packet prioritization." Signals, Circuits and Systems (ISSCS), 2011 10th International Symposium on. IEEE, 2011.

[19] Yanhua Liu, Jie Jin, Zongsheng Lai, “A dynamic adaptive arbiter for Network-on-Chip”, Journal of Microelectronics,Electronic Components and Materials,Vol. 43, No. 2,pp. 111 – 118, 2013.

[20] Fan, Xiaoxin. "Fault diagnosis of VLSI designs: cell internal faults and volume diagnosis throughput." (2012).

[21] Saidu, Ibrahim, et al. "A load-aware weighted round-robin algorithm for IEEE 802.16 networks." EURASIP Journal on Wireless Communications and Networking 2014.1 (2014): 226.

[22] Dally, William James, and Brian Patrick Towles. Principles and practices of interconnection networks. Elsevier, 2004.

[23] Berhanu, Yosef, Abebe Alemu, and Manish Kumar Mishra. "Dynamic Time Quantum based Round Robin CPU Scheduling Algorithm." International Journal of Computer Applications 167.13 (2017).

[24] Bedre, Abhilash L., and V. Nithish Kumar. "A Hybrid arbiter to accelerate performance of high speed soc." 2017 International conference on Microelectronic Devices, Circuits and Systems (ICMDCS). IEEE, 2017.

[25] Mahalakshmi, S. Sundara, and S. Cammillus. "Evaluation of Arbitration Techniques for SoC Design." *Evaluation* 4.4 (2017): 212-215.

Afshan Amin Khan received B.Eng. Degree in ECE from Islamic University of Science and Technology, India in year, 2011 and M .Tech Degree in ECE from Lovely Professional University, India in year, 2014. He is currently a Research Scholar in the Department of CSE at National Institute of Technology, Srinagar, India. His research interests are in the field of scheduling algorithms, NoC and SoC design.

Roohie Naaz Mir is a Professor & HoD in the Department of Computer Science & Engineering at NIT Srinagar, INDIA. She received B.E. (Hons) in Electrical Engineering from University of Kashmir (India) in 1985, M.E. in Computer Science & Engineering from IISc Bangalore (India) in 1990 and Ph.D from University of Kashmir, (India) in 2005. She is a Fellow of IEI and IETE India, senior member of IEEE and a member of IACSIT and IAENG. She is the author of many scientific publications in international journals and conferences. Her current research interests include reconfigurable computing and architecture, mobile and pervasive computing, security and routing in wireless adhoc and sensor networks.

Najeeb-ud-din received the B.E. degree in electronics and communications engineering from the Kashmir University, India, in 1985, the M. Eng. degree in solid-state electronics from the University of Roorkee, India, and the Ph.D. degree from the Indian Institute of Technology (IIT), Bombay, India, in 2003. He is currently a Professor in the Department of Electronics and Communication Engineering, National Institute of Technology, Srinagar, India. His research interests are in the field of SOI, Ferroelectrics, Organic Semiconductors, CMOS devices, design, and technology, CMOS in mixed-signal applications, and Reconfiguration.