Performance of Low Buffer Resource Flexible Router for NoCs

Israel Mendonça dos Santos
and Felipe M. G. França
PESC/COPPE, Universidade Federal do Rio de Janeiro
Rio de Janeiro, RJ, Brasil
Email: {israel, felipe}@cos.ufrj.br

Victor Goulart
E-JUST Center, Kyushu University
Fukuoka, Japan
Email: goulart@ejust.kyushu-u.ac.jp

***Abstract*—Nowadays, the performance of advanced multi-core systems is mostly limited by communication bottlenecks instead of computing speed or memory resources. Interconnection networks start to replace buses as the standard system-level interconnection infrastructure. Many researchers have been working on on-chip interconnection networks (Network-on-Chip - NoC) to improve its performance in terms of latency, throughput and power consumption. Buffers are the most influential element affecting performance in this architecture, and also the most expensive resource. The vast range of applications concurrently executing in the network and their different communication patterns leave some buffers of the routers underutilized. In this paper, we study the performance of low buffer resource flexible routers which can adaptively schedule resources (buffers) according to the dynamic behavior of the communication demand in the NoC. Our simulations showed this router architecture was able to improve throughput up to 21% or have similar performance while using half the number of buffers compared to a standard router.**

***Keywords*—Flexible Routers, Network-on-Chip (NoC), Adaptive Router, Buffer Resources.**

I. INTRODUCTION

Processor technology scaling down allowed the integration of billions of transistors on a single chip, and the emergence of multi-core systems with dozens of hundreds of processors and multiple (distributed) memories. Such high complexity systems on chip are hard to implement with traditional bus-based communication infrastructure. To solve this problem the concept, of Network-on-Chip (NoC) [1] [2] [3] was introduced and is the focus of recent research. In a NoC-based system, on-chip modules exchange code or data using a network as a sub-system for the information traffic. NoCs are built from multiple point-to-point data links interconnected by routers. Recently, researchers have been pushed to substitute the bus architectures by NoCs due to their scalability and efficiency benefits. In the design of NoCs, high throughput and low latency are both important design parameters and the router is the essential key to meet these requirements. High-throughput routers are required to allow an outflow of packets that matches the needs of the applications. Normally, a higher number of buffers improves performance (network throughput), but they also impact the power consumption, being responsible for about 64% of the total router leakage power [4].

NoCs can have different communication performance and costs, depending on their architecture features and design, and their target applications. Designing the same NoC router to cover the whole spectra of applications would lead to an oversize and expensive router, in terms of area and power. On the other hand, if one tries to design an application specific router for different markets, the designer would need to take several design decisions in a very early stage, eventually compromising scalability and optimization capabilities for distinct application behaviors.

According to Dally and Towles [5], a router role lies basically in efficiently traversing packets through the network links. Router buffering is used to hold packets that are unable to be forwarded to the desired output port due to contention. An ideal router should be able to adapt to different application requirements at runtime without compromising its performance. According to the experiments realized by Xuning and Peh [4], even at high loads, there were still 85% of idle buffers, which highlights the importance of performance optimization. In fact, it has been observed that storing a packet consumes far more energy than transmitting if forward [6].

Given the aforementioned significance of the NoC buffers, the concept of Flexible Router [8] [9] was introduced; it has a new router architecture approach that fully utilizes the available buffers in a flexible or adaptive way. When all the buffers of an input port are already in use and the upstream router asks for a buffer space, instead of just denying the request, and so causing contention, the flexible router will try to allocate any other empty buffer to the upstream, and, by doing so, avoiding contention.

Flexible Routers are a potentially promising architecture approach to overcome buffer under utilization and improve performance. In this work, a router that can reconfigure the buffers to better attend the communication demands is analyzed under various architecture design parameters. The result showed that the proposed router improved over the traditional router's performance in terms of throughput up to 21% and having similar performance when using half of the buffers.

This paper is organized as follows. Section II gives background information about the traditional baseline router and summarizes the related works. The Flexible Router architecture and functionality is explained in Section III. Section IV shows the experimental framework and results. Finally, the conclusion is shown in Section V.

 ISBN: 978-1-61208-368-1

II. Background and Related Work

Before explaining the functionality of the flexible router and its advantages, we present a brief introduction about the operation of the base router and the state-of-art concerning it.

A. Baseline Router

The baseline router used in this paper uses the architecture proposed by Dally and Towles [5], Figure 1. It is composed of several components that collectively implement the routing function and flow control functions. These functions are required to store and forward flits en route to their destinations through the network. The components are input controllers, one or more buffers per input or output port, a routing component, an arbitration component, and output controllers. A base router usually implements wormhole flow-control [10] with P ports, where P depends on the dimension of the topology. In a 2-D topology, P=5, as it includes the 4 ports to the neighbor (North (N), South (S), East (E), and West (W)) and the local port (L) (from/to the processor or memory core). Every input port has a certain amount of buffers to store flits. Flits are saved in buffers because they were unable to cross the network due to diverse factors, such as lack of credits on the link, lose the switch arbitration, among other things. The buffers are organized as *First In First Out* (FIFO) queues. Virtual Channel flow control [11] is the technique that separates the allocation of the buffers from the allocation of the channel. In case of blocking in one buffer, the input port can make use of another buffer. Flits entering the router are put in one of these buffers, called Virtual Channels (VCs), depending on their Virtual Channel Identifier (VCID). Queues on the input port are connected to a crossbar that links any input buffer to any output port.

The packet processing on the base router can be separated in a five-stage pipeline: The first stage is the Routing Computation (RC), where the Routing Module will decide an output port based on information extracted from the packet header. After RC, Virtual Channel Arbitration (VA) is done. During this stage, a buffer on the downstream router is selected. The VC Allocator sends a request signal to the downstream router that replies it with a grant signal containing a VCID (or a "no-buffer signal" depending of the availability of the buffer). Based on the answer of the downstream router, the requester upstream router will attach the replied VCID to every flit of the packet so that when it arrives in the downstream, it is put in the correct virtual channel. After the VA stage, we have the Switch Arbitration stage (SA). In this stage, the flits will compete for the usage of the crossbar. At the Switch Traversal (ST) stage the flits cross the crossbar in the direction to the output port. And, finally at the Link Traversal (LT) stage, the packet is forwarded to the next router.

The main limitation during the Base Router operation is the contention problem. Contention occurs when a buffer request at some input port is blocked because this port has no buffer to allocate this packet. Such contention may lead to further blockings in the network and hence congestion occurs [12] [13], which degrades the performance of the network.

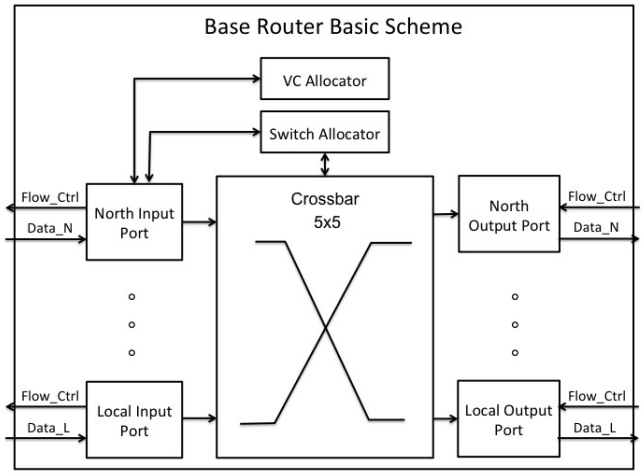

Fig. 1. Base Router Architecture [5].

B. Related Work

Several works have been proposed to maximize the performance through different buffer management techniques within the router. In Karol et al. [14] and Ramanujam et al. [15], a shared-buffer scheme is proposed to emulate an Output Buffered Router (OBR), but their mechanism greatly increases the number of the buffers of the router by adding extra Middle-Memories and a difficult control logic, which increases the area by 58% and the power usage by 35%.

In ViChaR [16], a Unified Buffer Structure (UBS) [17] is proposed with a dynamic virtual channel regulator called ViChaR, which also dynamically allocates virtual channels and buffers according to the network traffic conditions. They used the unified buffer unit instead of partitioned buffers as in our approach. The UBS is committed to create an illusion that the number, and size, of virtual channels of the router varies according to the traffic load. As the traffic load increases, ViChaR disposes more virtual channels to amortize the effect of the traffic. Although this mechanism has notable advantages, it has also suffered from a complex logic implementation increasing the area and probably causing power overheads.

In [18], Matos et al. propose the usage of a partitioned buffer that could lend/borrow buffer slots from each neighbor is proposed. By doing so, the depth of each channel could vary in time according to the traffic demands. But virtual channels were not used, only one buffer was available per channel, and since it could lend any buffer slot from its neighbors, it needed several multiplexers to provide the correct functionality of the mechanism, increasing the area and power demand. Also, the size of the multiplexer was determined by the size of the buffer, increasing linearly the necessary area. In our approach, the size of multiplexers is fixed, so the depth of the virtual channels have no influence on the size of our multiplexer. Also, our control mechanism is simpler, allowing the router to work in high frequencies, differently from the two previously mentioned approaches.

The idea of the flexible router was proposed in [8] and [9]. In this works, flexible router was implemented using Store-and-Forward (SAF) flow control and Virtual Channels were

 ISBN: 978-1-61208-368-1

not used. Due to this limited implementations, it was not possible to verify all the improvements that the flexible router architecture could provide. In this work we do a broader evaluation under different design parameters for the router, such as Wormhole flow-control, and use of Virtual Channels, analyzed under different traffic patterns, packet sizes and number and size of virtual channels.

III. Proposed Flexible Router

Statistically, not all buffers are used all the time, leading to underutilization of these valuable resources. In our architecture, it is possible to dynamically allocate idle buffers to other channels. The proposed Flexible router addresses the poor utilization of buffers which lead to lower than the theoretical ideal throughput. At the same time, it tackles the area constraints of implementing a shared buffer router. With a simple control mechanism and the addition of a module called FIFO Flexibility Controller (FFC), the virtual channels are decoupled from the input port, as now any flit is able to acquire any buffer provided that VC is not in use already. Essentially, FFC virtually provides more buffers allocation resources between routers.

To guarantee packet delivery, a flow control mechanism needs to be employed. The proposed flexible router uses on/off flow control in order to prevent the drop of packets. If the downstream router is about to get full, a control signal is sent to the upstream router; by receiving this message, the upstream knows that the VCs of the downstream are about to get full and stop transmitting flits until another message arrives informing that the downstream router buffers are free.

Since power is a valuable resource in NoC domain, the power-performance trade-off of variable size and amount of VCs configurations must be explored. Although, for some cases, the performance of the flexible router can be the same as the baseline router, we try to reduce the amount of resources of our router and still have reasonable performance. Therefore, we evaluate configurations in which the flexible router has less buffer space available than the baseline router and still achieve similar results.

Finally, on-chip routers ought to operate at high clock frequencies. It is a design need to carefully architect the pipelines with low complexity logic at each stage. Our design employs a balanced five-stage pipeline. The proposed architecture can achieve higher performance than the traditional router with comparable buffering while adding reasonable area and complexity overhead in managing the flexibility of the input ports.

A. Flexible Micro-architecture and Pipeline

Figure 2 shows the router micro-architecture of the proposed flexible router and its corresponding five-stage pipeline. Incoming flits are first stored in their respective input buffer, which are segmented into a certain amount of VCs. The RC stage of the flexible router, differently from the base router, employs a look-ahead routing scheme, where the output of a packet is computed based on the destination coordinates, one hop in advance. This happens only for the head-flit of the packet. This is done so that the FFC mechanism can use this information to avoid deadlocks. Further explanation

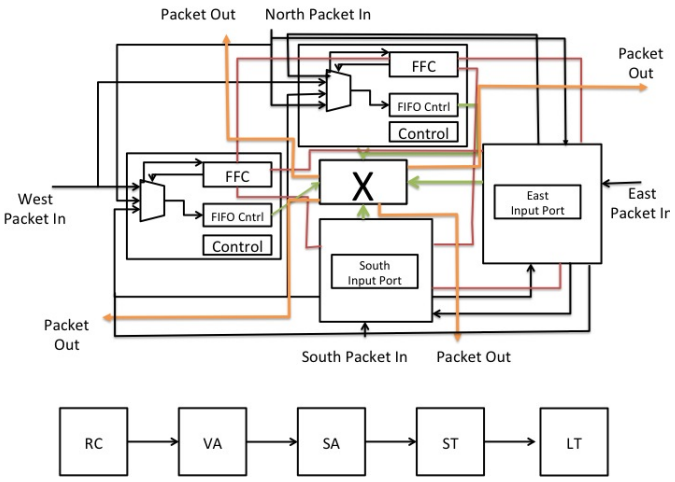

Fig. 2. Flexible Router Architecture.

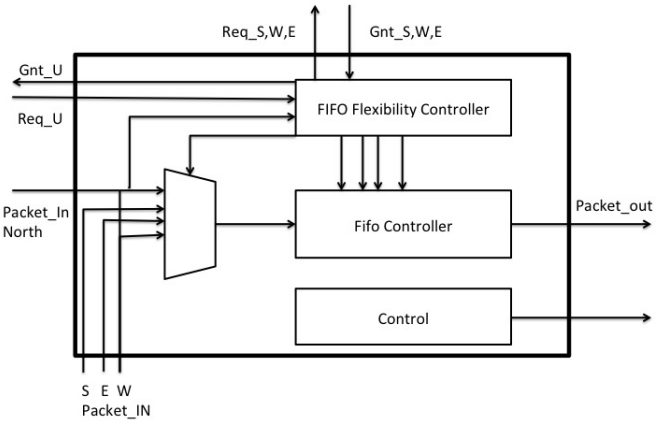

Fig. 3. Modified input port. A multiplexer and an extra module called FFC was employed.

about deadlocks will be provided in subsection B. The VA stage is substantially different from the base router. Instead of arbitrating for free VCs in a specific input port at the downstream, it can be extended to the other buffers belonging to other input ports if they are not being used. After the VA stage, the router pipeline is exactly the same as the one on the baseline router. Subsequently, the packets dispute for the use of the crossbar on the SA stage. Winning flits of the SA stage traverse the crossbar at the ST stage, and finally the flit departs from the output port at the LT stage.

Figure 3 shows proposed input port modules of the North Channel as an example. In this architecture, it is possible for a channel to lend its idle buffers to other channels. In conformity with this figure, each input port now has an extra multiplexer and an extra unity called FFC. The multiplexer is required in order to deviate the packets from their original channel and store them in the North buffer. The FIFO Flexibility Controller deals with the virtual channel allocation. Besides allocating VCs in its respective input port, it also works by allocating free

 ISBN: 978-1-61208-368-1

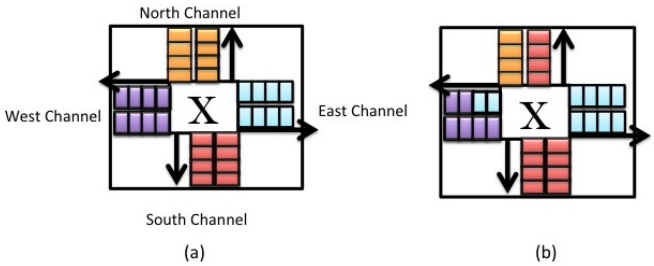

Fig. 4. (a) Baseline router designed with two buffers per port and FIFO depth 4; (b)Flexible router with the buffers reconfigured to attend the demand.

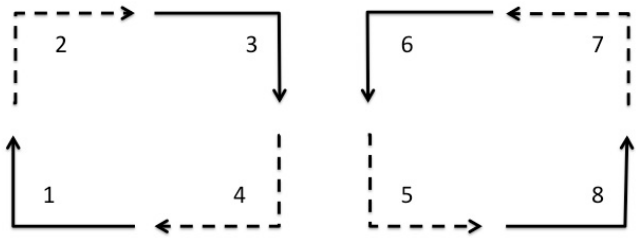

Fig. 5. Eight possible turns from a packet within a router.

virtual channels that belong to other input ports. Whenever a header flit enters the VA stage at the upstream router and issues a request to the FFC of the downstream one, this request will be forwarded to the FFC in the downstream router. FFC will preferentially first look for free VCs into its corresponding input port, if it fails to do so, it will try to reserve a VC belonging to another input port. The process of reserving a VC at the neighbor input ports is an asynchronous process with three stages started whenever a FFC detects that it will not have enough buffer space to house the incoming flit. The first stage of the process starts when it issues a request to other FFCs. In the second stage, the FFC that receive the request message, will lookup into its virtual channels and if it finds an empty VC, a reply signal is sent with an ACK (if there are no free VCs, it will send a NACK). The last stage of the communication is when the issuer of the request message receives the replies from its neighbors. Then it makes a selection between the ACKs and replies the VCID to the upstream router (in case of receiving just NACKs, it replies a NOT AVAILABLE message).

To keep track of the availability of the VCs, we created an auxiliary structure called the Flexible Availability Table. The table contains the identification of each buffer and its status. If the FFC reserve one buffer, it is marked in the table as *reserved*. At the end of the usage of this buffer, the FFC changes the status in the table to *available*. This kind of structure reduces the amount of communication because the FFC asks directly to a port that is known to have empty VCs.

In this design, the usage of local input port buffers is not considered. The flexible router local port is not different from the one in the baseline router. Only the North, South, West, and East ports were changed. This is done in order to make the Flexible Router mechanism simple and not affect the injector queues.

The policy of choice of the neighbor VC, in case of more than one ACK, may influence the router performance. In our router, it was employed Dimension Order XY (DOR X-Y) algorithm [7], so the probability of filling up the buffers of the X direction is greater than the ones in the Y direction. Considering this fact, the Buffer Choosing Algorithm tries to choose buffers from ports that have less probability to be used through the execution of the system. The algorithm will be better clarified in subsection B.

Figure 4 shows how the VCs of the flexible router can be reconfigured. First, the depth of the buffers must be defined at design time, in this example, it has size equal to 4 and the packet size is 2, as illustrated in Figure 4(a). After this, the traffic is unbalanced and an entire virtual channel from North is lent to South port, and two buffer slots from West port are lent to East port, as shown in Figure 4(b). One can look that even though the West port houses a packet from the East port, it also can store packets from other ports as well. Flexible router has no issue about mixing packets from different ports in the same buffer, the only thing to be taken into consideration is not to interleave the packets. The mix is possible because once the packet acquires a VC, the routing information of the reader will be used for all the following packets of the VC until a tail flit is founded. If we interleave packets, body flits would go to wrong routes. The allocation of several packets in the same input port is possible because the tail flit releases the VC when it leaves the router. By doing so, even if the buffer is not empty, it can be reallocated to another packets if the previous packet is totally inside the buffer. We do not allow flits to go to different buffers because it would add complexity to organize these flits. The basic mechanism of the FIFO is thus kept.

B. Deadlock Problem

Due to the flexibility of the flexible router, any packet can be stored in any buffer regardless of its direction. This situation may lead to deadlock. For example, considering two neighbor routers connected through a channel in the X-axis (A and B), if all packets stored in router A are going to the Eastern direction and, at the same time, all packets in router B are going to the Western direction, a deadlock will occur, since there will be no available space in the destination buffers, and vice versa. This problem can be extended for more than two routers. The solution for this problem is analogue to the one used in [8]. As illustrated in Figure 5, there are eight possible turns that a packet can do inside a router. The turn-model [19] restricts the usage of two out of the eight turns. By restricting turns, the turn model imposes a total order on the allocation of the buffers, hence avoiding deadlocks.

C. Virtual Channel Allocation Method

Due to the aforementioned deadlock problem, a special Virtual Channel Arbitration algorithm must be implemented. This algorithm will implement the restrictions and will also give priority to allocate buffers in the vertical direction. Since we are using a X-Y algorithm, buffers of the X axis (West and East) might become heavily utilized. So, it makes sense to give priority to the allocation of the routers belonging to the Y axis (North or South).

 ISBN: 978-1-61208-368-1

The proposed mechanism consists of lending VCs from other input ports according to the availability of the buffers. If a packet wants to be transmitted and the input port is full, the FFC makes a request for an empty buffer in other ports. At every cycle, FFC look up the table trying to find an available buffer. If it fails, a request will be placed to another FFC. Even though the VC is available, it doesn't guarantee that it will be reserved to the requesting FFC; the same buffer can be requested for more than one FFC, in this case the owner of the buffer will choose a winner using a round-robin arbiter. In the buffer availability check, the FFC request and reply can be performed within one cycle, which means performance overhead. Moreover, as the channel demands for each router can vary in time for one or multiple applications, the buffer allocation of the flexible router is dynamic at *runtime*.

IV. Experiment Configuration and Analysis

A. Simulation Platform

To evaluate the efficiency of the proposed flexible router against the baseline router, a cycle-based accurate simulator called TOPAZ [20] was employed. TOPAZ is implemented in C++, models the pipeline of the routers, and operates at the granularity of individual architectural components. Simulations were performed using 64 routers organized as a 8x8 MESH network. In the simulator, we varied the buffer size, packet size, and quantity of virtual channels per input port over different traffic loads. Table I summarize the simulation parameter evaluation.

Each router has five physical bi-directional channels (ports) including the local port. The simulator keeps injecting messages into the network until it reaches 50.000 messages. A simple DOR-XY routing is used for all of our simulations where packets are first routed in the X-dimension followed by the Y-dimension. A DOR-XY was used because the main focus is on highlighting the improvement in performance due to the flexible router's adaptability, rather than the routing algorithm's. Wormhole flow control is also employed to control the flow of packets alongside the network, complementing previous evaluations. The tested network traffic pattern was the Uniform Random, where a node injects messages into the network at regular intervals specified by the injection rate. Normal Random distribution, where each node has the same probability of being chosen as a destination for a packet, was used for the destination node selection.

B. Analysis of Results

Our simulation exploration starts with the throughput comparison between the conventional, statically assigned buffer architecture in the base router compared to the proposed flexible router implementation. We started our discussions assuming that both use the same number of virtual channels. Results are shown varying the packet size from four flits up to sixteen flits for both of them. The injection load measures how many flits are injected in the network per cycle per router (flits/cycle/router), and the throughput is the amount of flits that leave the network per cycle (flits/cycle). Each line is labeled after the architecture in use and the packet size. For example, Flex 4 means that the line shows the performance of the scenario using a packet size of 4 and the flexible router architecture.

As the injection rate grows, the difference in throughput between the flexible and the base router increases. From the figures 6 to 8, we can see the difference for several buffer sizes using two virtual channels. It can be noticed that the difference in throughput grows up to 21% (4 flits per buffer, Figure 6) and the flexible router outperforms the base router in most cases 9% (8 flits per buffer, Figure 7) and 11% (16 flits per buffer, Figure 8). Another important result is that flexible saturates for higher injection loads when compared to the base one. The difference between both approaches is reduced as more resources are introduced to the network (larger buffers and increased amount of virtual channels). This is expected since the flexible router more efficiently manages routers with more limited number of resources. As the number of virtual channels grows, the flexible router starts to show a closer performance to the base router's. For four virtual-channels, the difference in performance between the two routers is small. Using packet size of 16 flits, flexible router outperforms the base router in 6% for buffer size of 4 and 8 flits (Figure 9), and 3% for buffer size of 16 flits (Figure 10 and Figure 11). Even though the flexible router outperform the base router in most cases (for packets of 4 flits and buffer size of 8 and 16 flits), the baseline router has respectively 10% and 8% better performance (Figure 10 and Figure 11).

As we can see in the charts, the best results are the ones in which the packet size matches exactly the size of the buffer size. An explanation for this is that packets can be moved entirely through VCs. By doing so, a packet doesn't interfere with other packets in the same buffer since no packets are allocated together. As the size of the packets grows, one needs more than one buffer to allocate these packets, in the worst case needing to allocate four buffers in different routers when the packet has sixteen flits and buffer size is four. The lack of resources increases the congestion of the network.

Another observation made in this paper is the performance of the flexible router using two virtual channels when compared with the performance of the baseline router using four virtual channels and both using four flits per buffer in an 8x8 mesh. As we can see in Figure 12, even though the baseline router has doubled the number of buffers, the throughput of the flexible router is only 3% inferior. When we added more two buffers in the baseline router, the contention problem was reduced improving its overall performance, but the flexible router was also able to overcome the same level of contention using just half the number of buffers. This implies great reductions in router area. As mentioned before buffers occupy most of the area of the router. This reduction is not only in the area but also in energy since the buffers are power-hungry components. A detailed discussion about the energy and the area savings caused by a flexible router can be found in [21].

Due to the inherent capacity of the flexible router to allocate resources in a more efficient way, the average bandwidth of the network is increased, which also increases the outflow of packets, and by doing so, mitigating the congestion problem.

V. Conclusion and Future work

In this paper, a perform analysis was done on the effects of the usage of virtual channels of flexible routers, as much as the employment of a wormhole flow control, which improved the

 ISBN: 978-1-61208-368-1

TABLE I
SIMULATION PARAMETER VARIATION

Parameter	Values
Packet Size	4, 8, 12, and 16
Buffer Size	4, 8, and 16
# of Virtual Channels	2 and 4

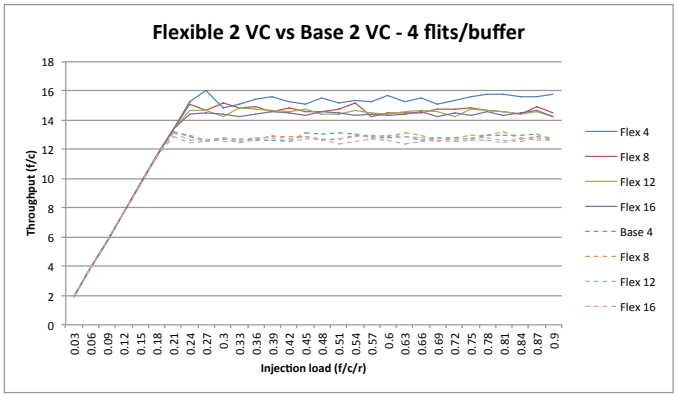

Fig. 6. Throughput - 2 Virtual Channels and 4 flits per buffer.

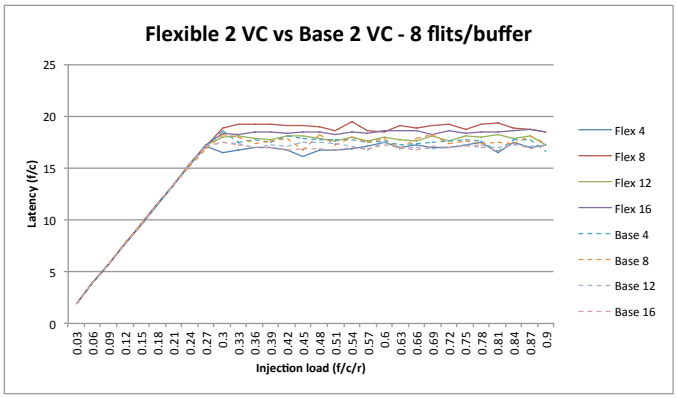

Fig. 7. Throughput - 2 Virtual Channels and 8 flits per buffer.

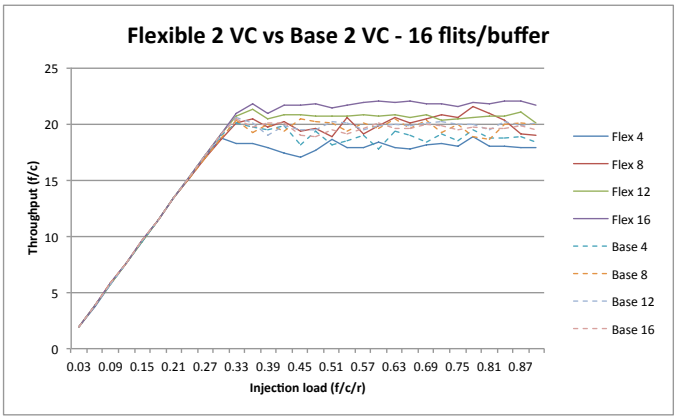

Fig. 8. Throughput - 2 Virtual Channels and 16 flits per buffer.

utilization of buffers proposed by the original flexible router and consequently enhanced its throughput and latency. Due to the before mentioned improvements, the contention problem

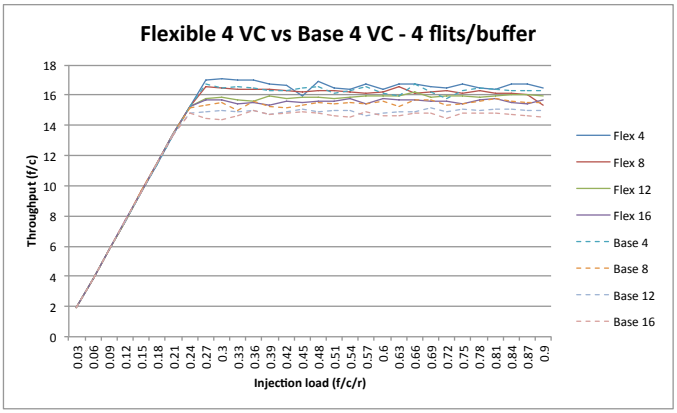

Fig. 9. Throughput - 4 Virtual Channels and 4 flits per buffer.

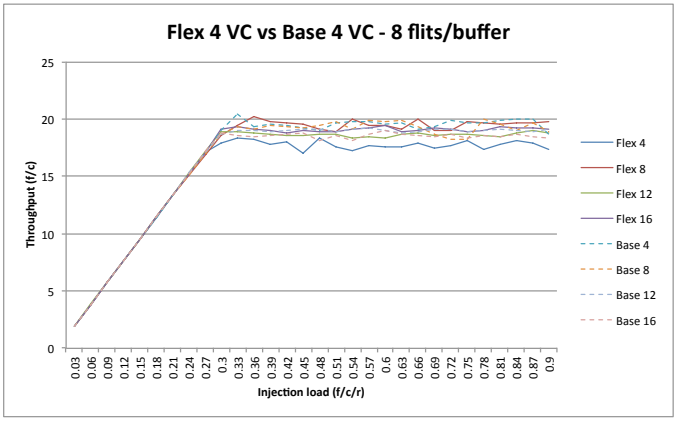

Fig. 10. Throughput - 4 Virtual Channels and 8 flits per buffer.

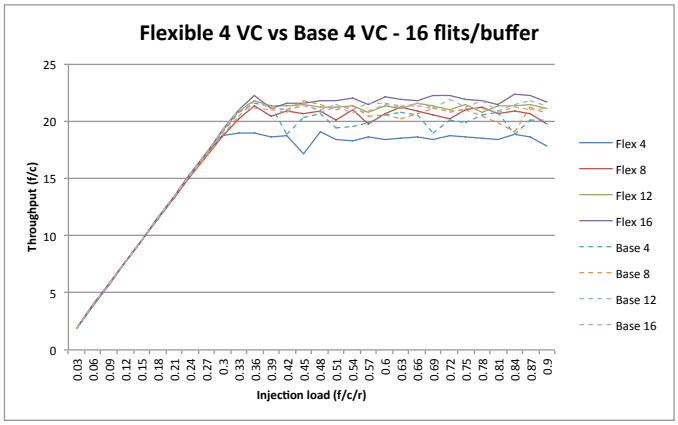

Fig. 11. Throughput - 4 Virtual Channels and 16 flits per buffer.

was smoothed and the performance was increased up to 21% when buffers are small.

A flexible router was shown to be more efficient when the resources of the network were limited and had similar performance of the base router with double resources. Having presented the achievable improvements done by the flexible router, the new router architecture should be considered as an alternative for the base router when the network lacks resources when under heavy injection loads.

 ISBN: 978-1-61208-368-1

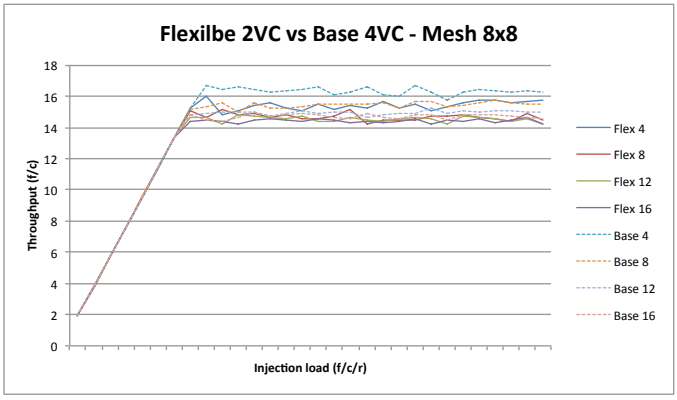

Fig. 12. Throughput - 2 Virtual Channels flexible and 4 Virtual Channels base.

As future work, we intend to test new routing approaches, traffic-patterns and workloads and traces from existing System-on-Chip architectures.

ACKNOWLEDGMENT

The authors would like to thank Coordenação de Aperfeiçoamento Pessoal de Nível Superior (CAPES), Brazil, and Japan Student Services Organization (JASSO) Foundation for funding Israel M. Santos' work.

REFERENCES

[1] L. Benini and G. D. Micheli, "Networks on Chips: A New SoC Paradigm," IEEE Computer, vol. 35, pp. 70-79, 2002.

[2] A. Hemani, A. Jantsch, S. Kumar, A. Postula, J. Oberg, M. Milberg, and D. Lindqvist, "Network on chip: An architecture for billion transistor era," in Proceedings of the IEEE NorChip Conference, pp. 65-79, 2000.

[3] W. J. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," in Proceedings of the Design Automation Conference (DAC), pp. 684-689, 2001.

[4] C. Xuning and L. S. Peh, "Leakage power modeling and optimization in interconnection networks", in Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED), pp. 90-95, 2003.

[5] W. J. Dally and B. Towles, "Principles and practices of interconnection networks", Morgan Kaufmann, 2004 .

[6] T. T. Ye, L. Benini, and G. D. Micheli, "Analysis of power consumption on switch fabrics in network routers," in Proceedings of the 39th Design Automation Conference (DAC), pp. 524-529, 2002.

[7] Duato, J. , A new theory of deadlock-free adaptive routing in wormhole networks, IEEE Transactions on Parallel Distributed Systems, v. 4, n. 12, pp. 13201331, 1993.

[8] M. S. Sayed, A. Shalaby, M. E.-Sayed, and V. Goulart, "Flexible router architecture for network-on-chip". Computers & Mathematics with Applications, pp. 13011310, 2012.

[9] M. S. Sayed, A. Shalaby, M. Ragab, M. E.-Sayed, and V. Goulart, "Congestion mitigation using flexible router architecture for Network-on-Chip", Electronics, Communications and Computers (JEC-ECC), 2012 Japan-Egypt Conference on, pp.182-187, March, 2012.

[10] W. J. Dally and C. L. Seitz, "The torus routing chip,"Journal of Distributed Computing, vol 1(3), pp. 187-196, 1986.

[11] W. J. Dally, "Virtual-Channel flow control", in Proceedings of the 17th Annual Internation Symposium on Computer Architecture (ISCA), pp. 60-68, 1990.

[12] L. Benini and and G. De Micheli, "Networks on Chips: Technology and Tools", Morgan Kaufmann, 2006.

[13] M. J. Karol, M. G. Hluchyj, and S. P. Morgan, Input versus Output queuing on a space-division packet switch, IEEE Trans. Communication., Vol. 35, no. 12, pp. 1347-1356, December, 1987.

[14] R.S. Ramanujam, V. Soteriou, B. Lin , and Li-S. Peh, "Design of a High-Throughput Distributed Shared-Buffer NoC Router", Networks-on-Chip (NOCS), pp. 69-78, May, 2010.

[15] R. S. Ramanujam, V. Soteriou, B. Lin , and Li-S. Peh, "Extending the Effective Throughput of NoCs With Distributed Shared-Buffer Routers", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, v.30(4), pp. 548-561, April, 2011.

[16] C. A. Nicopoulos, D.Park, J. Kin, N. Vijaykrishnan, M. S. Yousif, R. Das Chita, "ViChaR: A Dynamic Virtual Channel Regulator for Network-on-Chip Routers". 2006 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO06), pp. 333346, December, 2006.

[17] Y. Tamir and G.L. Frazier, "High-performance multiqueue buffers for VLSI communication switches", in: Proceeding of the 15th Annual International Symposium on Computer Architecture, ISCA, pp. 343354, May, 1988.

[18] D. Matos, C. Concatto, F. Kastensmidt, L. Carro, A. Susin, and M. Kreutz, "Reconfigurable Routers for Low Power and High Performance", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, pp. 2045-2057, September, 2010.

[19] C.J. Glass, and L.M. Ni, "The turn model for adaptive routing", in: Proceeding of the 19th Annual International Symposium on Computer Architecture, ISCA, pp. 278287, May, 1992.

[20] P.Abad, P.Prieto, L.Menezo, A.Colaso, V.Puente, and J.A. Gregorio, "TOPAZ: An Open-Source Interconnection Network Simulator for Chip Multiprocessors and Supercomputers", Networks on Chip (NoCS), 2012 Sixth IEEE/ACM International Symposium on, pp. 99-106, May, 2012.

[21] H. El-Sayed, M. Ragab, M. S. Sayed., and V. Goulart, "Hardware Implementation and Evaluation of Flexible Router Architecture for NoCs", in Proc. of 20th IEEE Intl. Conf. on Electronics, Circuits and Systems, pp. 621-624, December, 2013.

 ISBN: 978-1-61208-368-1