QOS DRIVEN NETWORK-ON-CHIP DESIGN FOR REAL TIME SYSTEMS

Ankur Agarwal,
CSE Dept. FAU,
Boca Raton, FL 33431
email: ankur@cse.fau.edu

Mehmet Mustafa,
Verizon Laboratories,
Waltham, MA 02451
email: mm01@rcn.com

A. S. Pandya
CSE Dept. FAU,
Boca Raton, FL 33431
email: pandya@fau.edu

Abstract

Real Time embedded system designers are facing extreme challenges in underlying architectural design selection. It involves the selection of a programmable, concurrent, heterogeneous multiprocessor architecture platform. Such a multiprocessor system on chip (MPSoC) platform has set new innovative trends for the real-time systems and system on Chip (SoC) designers. The consequences of this trend imply the shift in concern from computation and sequential algorithms to modeling concurrency, synchronization and communication in every aspect of hardware and software co-design and development. With a billion transistors era, some of the main problems in deep sub-micron technologies characterized by gate lengths in the range of 60-90 nm arise from non scalable wire delays, errors in signal integrity and un-synchronized communication. These problems have been addressed by the use of packet switched Network on Chip (NOC) architecture for future SoCs and thus, real-time systems. Such a NOC based system should be able to support different levels of quality of service (QoS) to meet the real time systems requirements. Thus, it becomes extremely critical to properly design a network interface (NI) and the communication backbone for NOC. In this paper we present a component based design of network interface and communication backbone which supports different levels of QoS. The design has been tested for an adaptive wormhole routing with proactive turn prohibition to guarantee deadlock free on chip communication for NOC architecture. In this work we propose to use the modified turn prohibition (MTP), which has been shown to perform better than up/down and the turn prohibition.

Keywords: *Network-on-chip, network interface, quality-of-service, real-time-systems*

1. Introduction

Motivated by increasing needs for concurrent computation requirements for embedded systems, and advances in deep sub-micron technologies, the integration of an entire system onto a single die has become technically feasible, giving rise to the system-on-chip (SoC) era [1]. The System Level Design era where creativity, innovative ideas, ingenuity and inspiration come to the fore, is approaching and pushing technology at every turn. Moore's law predicts that a chip in 2010 will count more than four billion transistors operating in multi GHz range [2] [3]. It is expected that the future Systems-on-chip (SoCs) will integrate from several dozens to hundreds of cores, making huge amounts of computation power available in a single billion transistor chip. These cores can be memory banks, I/O blocks, general purpose programmable processors, DSP processors, co-processors or dedicated hardware blocks. This is mainly due to the exponential decrease in the transistor size enabling faster transistor switching times and more densely integrated circuits. Such computation power is required to support complex multimedia algorithms (for 3-D video, gaming, and other applications) and communications algorithms onto handheld systems. It is further assumed that in future embedded systems, memory will occupy more than 60% of the total hardware block area [2]. Such computation power has posed some challenges: gate delays have been constantly scaling down, while global wire delays typically have either increased or remained constant as repeaters are inserted [4]. It is estimated that in 50nm technology, global wire delays will reach up to 6-10 clock cycles [5] [6]. As a result, achieving synchronization onto the system will be very difficult if not unfeasible. System designers also need to keep the power consumption of the system at a manageable level. Under such considerations, a single-processor implementation will not suffice, thus driving the development of more and more complex multi-processor SoCs (MPSoCs) [7]. Another dominant factor is to be able to design the system in an acceptable timeline known as time-to-market. Also, system level designers are constantly looking for ways to support a set of demanding Quality of Service (QoS) parameters and performance metrics, as customers became more savvy.

Technology scaling at the same time has unwanted side effects which include cross coupling, noise, and transient errors [1]. This will raise further reliability concerns for system-level designers. This has again led us to reuse of design blocks, sometimes referred as components, which have been carefully designed by expert designers. However it can never be guaranteed that those sub-micron effects will not pop-up again while reusing those components in a design of a sub-system or a system. Thus it can be concluded that components or the sub-system which perform as expected might not perform in the same way after system integration [8]. This has led to a new domain of research work for system level integration and verification viz, the Network-on-Chip (NOC) architecture [9, 10].

It can be realized from the ITRS graph [11] that the manufacturing non recurring expense (NRE) of the chip with RTL design methodology alone would have been enormous

had future improvements not come about. In the past few decades, it is due to such improvements that teams of engineers and managers were able to bring the cost of a product development down to an affordable price for the customers, while enhancing quality of service and customer support relative to previous releases. However, if such innovations and future trends are not brought into the early stage of the product development cycle, the NRE cost of the product can increase to an unaffordable amount of one billion dollars by 2010 [11]. Thus it is expected that the future systems will have increasing roles of design automation, reusability and componentization, thus increasing the market share for the electronic design automation (EDA) Industry. For such scenario, System-level modeling environment should be developed that essentially supports the middle-out design philosophy to exploit reuse to the maximum in order to reduce the design effort [12]. The high volume of reuse should cut down the overall system design cycle [13].

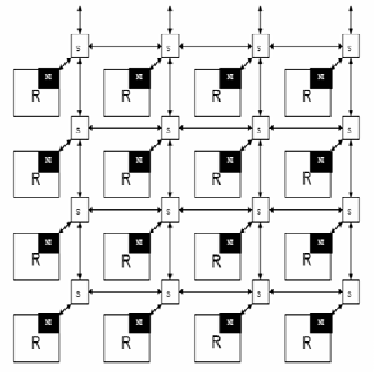

Figure 1. NOC Architecture

In traditional systems all the resources share one or more common buses and thus the same bandwidth. As a consequence, with the linear increase in the number of the processors onto the bus, the system performance decreases exponentially [14]. At the same time, with the advances in technology, it can be realized that gates relatively cost less than the wires. However, many innovations have been introduced in the design of bus architectures. These include pipelining, split-and-retry techniques, removal of tri-state buffers and multi-phase clocks, and various efforts to define the standard communication sockets [15]. However, in many cases introducing new bus architectures such as AMBA, ASB to AHB2.0, AMBA-lite and AMBA AXI, has required many changes in bus implementation, and more importantly bus interfaces, thus impacting IP reusability. Another reason that the buses are not scalable is that they cannot decouple the activities of the transaction, transport and physical layers [9].

On the other hand, network based communication strategies provide a reusable, scalable and highly flexible solution to coup with the current technology trends. Thus, most of the future systems would have several SoCs that will use network architecture and a packet based communication protocol for on chip communication, referred as NOC. The NOC architecture is represented in Figure 1. In Figure 1, the larger "R" block represents a resource, which in turn may consist of memory, processor, cache memory resource interface network, etc., connected by a local bus based connection. The smaller "S" block in Figure 1 represents a NOC router and switch. In NOC different local regions (synchronous regions) would communicate with other synchronous regions by switches and routers. As a whole, the system would be asynchronous in nature.

NoC's can improve design productivity by supporting modularity and reuse of complex cores, thus enabling a higher level of abstraction in the architectural modeling of future systems [9], [10]. No delays are experienced for accessing the communication infrastructure, since multiple outstanding transactions originated by multiple cores can be handled at the same time, resulting in more efficient network resource utilization. However, given a certain network dimension (e.g., number of instantiated switches), large latency fluctuations for packet delivery could be experienced as a consequence of network congestion. Such delays would be unacceptable in real-time systems. There are two plausible solutions to this problem: (1) over-dimensioning the network to achieve the worst case scenarios for a definite traffic pattern; (2) providing priority levels to the traffic. The second solution is a cost-effective solution in terms of the power consumption. In this way real-time requirements can be met by providing priority to real-time traffic. Such real-time traffic can be guaranteed for its delivery to its destination by either reserving some paths for real-time data, or implementing priority-based scheduling criteria.

2. Network Interface

Network Interface (NI) is the network protocol block. It defines and sets the protocol specification to be employed by the on chip communication network. Figure 2 depicts the block level details of NI.

NI gets the traffic from resource (R). The resources for the NOC can be any general purpose processor core, memory, specified controller, FPGA, ASIC etc. The data traffic from the resource will come to one of the Ports. There are three ports shown in the Figure 2 are corresponding to three priority levels (or service levels) for the data traffic. The priority levels are customizable. The ports will then pass the data traffic to the Input Queues (3 queues for three levels of service) through Open Core Protocol (OCP) layer.

OCP aims at achieving the goal of design re-use. It defines protocols to unify all of the inter-core communication and a high-performance, bus-independent interface between IP cores or other resources that reduces design time, design risk, and manufacturing costs for SOC designs. The OCP transforms the resource making them independent of the underlying architecture and design of the systems in which they are used. OCP helps in optimizing the die area by configuring only those features needed by the communicating cores. OCP further simplifies system verification and testing by providing a firm boundary around each IP core that can be observed, controlled, and validated. Synchronous unidirectional signaling in the OCP produces simplified core implementation, integration, and timing analysis. Any on-chip interconnects can be interfaced to

the OCP rendering it appropriate for many forms of on-chip communications. These communications include (1) Dedicated peer-to-peer communications, as in many pipelined signal processing applications such as MPEG2 encoding and decoding (2) A simple slave-only applications such as slow peripheral interfaces (3) High-performance, latency-sensitive, multi-threaded applications, such as multi-bank DRAM architectures. For a core to be considered OCP compliant, it must satisfy the conditions dictated in the OCP specification. Data from the OCP layer gets stored into one of the input queues as per the priority of the data. The scheduler chooses the data based on the given queuing discipline. Queuing discipline for the scheduler is a customizable parameter.

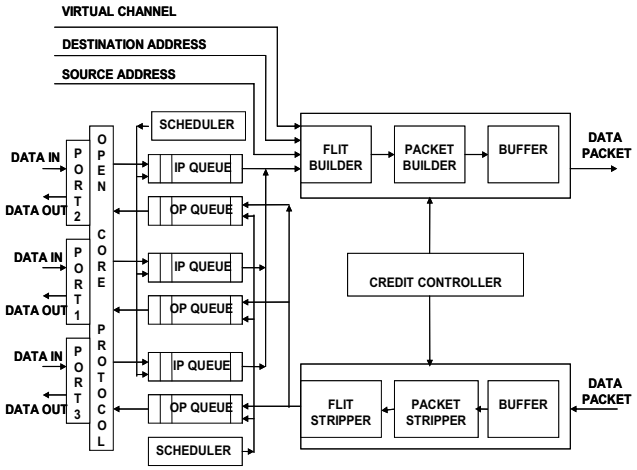

Figure 2. Network Interface Architecture

Data from these queues is sent to the flit builder. Along with the data field, the source address (SA) destination address (DA), and virtual channel (VC) information are also passed to flit builder block. SA is the address of the resource which passed the data through the ports. The DA is the address location where the data is finally sent to. The VC field refers to the number of the virtual channel through which this data will travel to the next node in form of flits. All flits corresponding to this data will travel through the same virtual channel. The flit builder makes a flit as per the specification defined in the Flit Structure part of the document. The flits are then passed to the packet builder block. Packet builder block takes the flits and prepares the packet. The packets are then buffered in a buffer from where are scheduled to their destination address through the communication backbone. The lower of part of Figure 2 refers to the part of the NI which behaves as a destination location for a packet. When the data packet reaches its final destination (Virtual Channel Router), it is then passed to the NI. NI de-packetizes the data and then passes it to the corresponding resource (R). The data flits are first buffered into the buffer, from where they are passed to the packet stripper block. The packet stripper removes the packet details and divides the packet in form of flits. These flits are then passed to the flit stripper part. The flit stripper part removes the flit details such as VC, SA, DA etc, and passes the data stream to one of the output queues. The data is sent to the corresponding output queue as per the priority level of the data packet. The scheduler chooses one of the output queues and forwards it data to the resource. The scheduling discipline is the user defined parameter such as round-robin, first-in-first-out or priority based. In this report we are assuming priority based scheduling. The credit controller block controls the end-to-end flow control by keeping track of the number of credits in the virtual channel buffer of the router circuit. Number of credits corresponding to a virtual channel signifies the amount of space for holding the number of flits. When a flit is forwarded to the virtual channel router through the buffer, one credit corresponding to the particular VC is reduced. In a similar way, once we receive a flit from the virtual-channel-router in the NI, one credit is added to the corresponding virtual channel. If the credit count of any virtual channel becomes zero, then the NI stops sending the flits to that particular VC until sufficient number (at least one) of credits are available again.

3. Flit Structure & Virtual Channel Router

In this section we discuss the wormhole router with two virtual channels and the flit structure. We assume that there is one flit-wide physical communication channel between adjacent routers. To make most effective use of the communication channel, we use the virtual channel bit in the flit that identifies which virtual channel will be used. This permits interleaving of flits in the physical channel and reduces the header blocking delay [16]. Since the NI forwards messages to the router with the highest priority first, all that the router needs to do is to route the messages to the indicated virtual channel or the output port. Note that blocking would occur if the requested output virtual channel is in use. In the router, the flit is first stored in one of the two input flit buffers as determined by the V bit. When the destination address flit is the flit buffer, router determines which physical output channel must be used. This could be done with a small look-up table in the router. In this arrangement, both the input and output channels have flit buffers; one for each virtual channel and the necessary multiplexer de-multiplexer circuits. As shown in Figure 3, input channel is de-multiplexed into either upper or lower flit buffer depending on the state of the V-bit in the flit. Since the state of the V-bit dictates which virtual channel will be taken on the output port, if the required virtual channel is busy, the incoming flit will be blocked and it will be waiting at the input flit buffer.

Wormhole environment is prone to deadlocks [17], [18], [19] and special measures must be taken to prevent deadlock formation. It was proved [17] that the absence of cycles in the channel dependency graph (CDG) is a sufficient condition for deadlock-free routing. It was later shown [20] that this is also a necessary condition for deadlock-free coherent routing algorithms. Authors in a series of publications [21], [22], [23], [24], [25] have shown that elimination of cycles in the channel dependency graph can be eliminated with a turn prohibition algorithm which operates on the network topology graph instead of the CDG. This algorithm guarantees minimality of fraction of prohibited turns that will not exceed 1/3 in any connected topology.

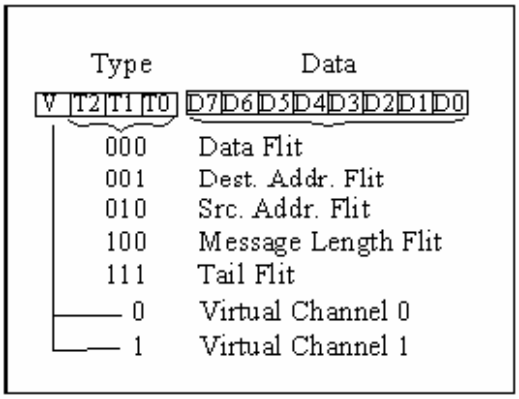

Figure 3. Flit Struture

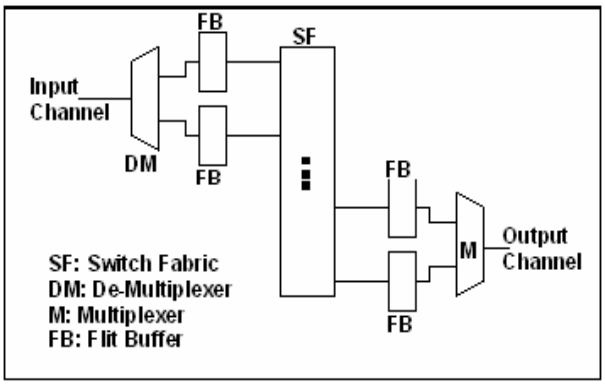

Figure 4. Virtual Channel Router

This algorithm is by no means the only turn prohibition algorithm in use. For example the spanning tree approach prohibits the use of non tree edges and therefore does not make efficient use of network resources. A more efficient variant known as the up/down approach also makes use of a spanning tree in the topology, but permits use of cross links in such a way that all cycles are broken. In the up/down approach the fraction of prohibited turns is non-minimal and its performance cannot be guaranteed [26].

4. Simulation Results

In this work we propose to use the modified turn prohibition (MTP) as described in [25], which has been shown to perform better than up/down and the turn prohibition [21]. One measure of the effectiveness of turn prohibition is the average dilation, which is defined as the ratio of the average distance in a topology with prohibition to the average distance without prohibition. In Figure 5 we show the average distance after each of the MTP, the TP, and the Up/Down approaches have been applied to the family of topologies. We show the average distance as a function of the bisection widths of topologies studied. Each experimental point in the plot corresponds to the average distance of 100 different topologies with the given bisection width. All nodes in the topologies studied had average degree of four. We also show the average distance in the same topologies without any turn prohibition.

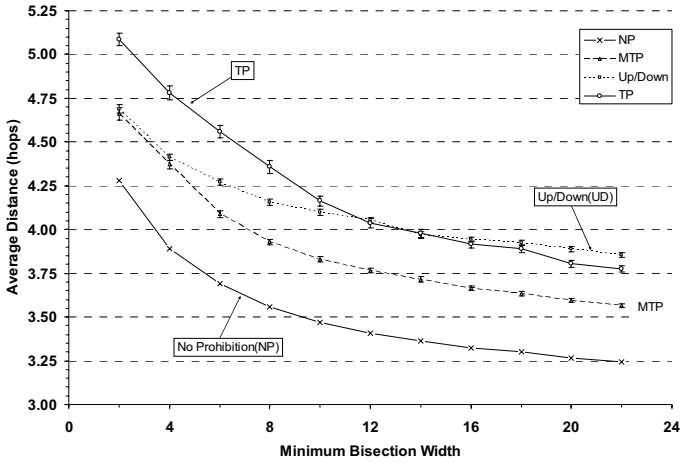

Figure 5 Experimental Results Comparing Average Distances

We see that for all of the topologies studied, the MTP algorithm [27] introduced the minimal dilation. In Figure 6 we compare the performance of the three algorithms with single virtual channel version of the wormhole router. In these simulations all flits were eleven bits wide with 8 bits for payload and 3 bits for the flit type. All messages were 200 flits long. The vertical axis shows the minimal sustained throughput in terms of worms/node-second showing MTP algorithms outperforming both the TP and the Up/Down algorithms.

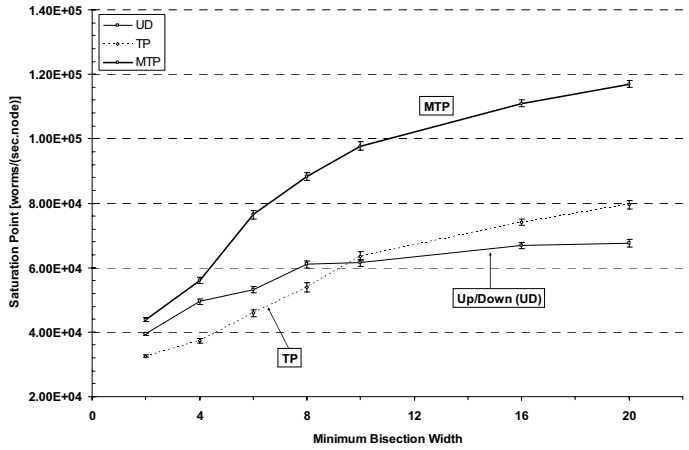

Figure 6. Saturation Points Comparing MTP, TP, and Up/Down Approaches in Single Virtual Channel Version of the Wormhole Router.

5. Conclusion

In this paper we propose to use a packet based communication for a multiprocessor based embedded systems. We propose to use the modified turn prohibition (MTP), which has been shown to perform better than up/down and the turn prohibition. in order to avoid the possible deadlock in inter-core communication. We show the average distance after each of the MTP, the TP, and the Up/Down approaches have been applied to the family of topologies. Each experimental point in the plot corresponds to the average distance of 100 different topologies with the given bisection width. All nodes in the

topologies studied had average degree of four. We also show the average distance in the same topologies without any turn prohibition. It can be analyzed that for all of the topologies studied, the MTP algorithm introduced the minimal dilation. We also provide the simulation results for the performance of the three algorithms with single virtual channel version of the wormhole router. In these simulations all flits were eleven bits wide with 8 bits for payload and 3 bits for the flit type. All messages were 200 flits long. The vertical axis shows the minimal sustained throughput in terms of worms/node-second showing MTP algorithms outperforming both the TP and the Up/Down algorithms.

References

[1] T. Das, C. Washburn, P. R. Mukund, S. Howard, K. Paradis, J. G. Jang, J. Kolnik, "Effects of technology and dimensional scaling on input loss prediction of RF MOSFETs," *International Conference on VLSI Design held jointly with 4th International Conference on Embedded Systems Design*, pp. 295-300, 2005.

[2] L. Benini and G. De Micheli, "Networks on chip: a new SOC paradigm," *IEEE Computer*, Volume 35, No. 1, pp. 70-78, January, 2002.

[3] Xu, Jiang, W. Wolf, J. Hankel, S. Charkdhar, "A Methodology for design, modeling and analysis for networks-on-Chip," *IEEE International Symposium on Circuits and Systems*, pp. 1778-1781, May 2005.

[4] S. Kumar, A. Jantsch, J-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani, "A network on Chip Architecture and Design Methodology," *In IEEE Computer Society Annual symposium on VLSI*, pp. 117-124, April 2002.

[5] A. Hemani, A. Jantsch, S. Kumar A. Postula, J. Öberg, M. Millberg, D. Lindqvist, "Network on Chip: an architecture for billion transistor era," *Proc. of IEEE NorChip Conference*, pp. 8, November 2000.

[6] P. Wielage, K. Goossens, "Network on silicon: blessing or nightmare?," *Euromicro Symposium on Digital System Design, Dortmund*, *Genmany*, Keynote Speech, September 2003.

[7] Jerraya Ahmed Meine, Wolf Wayne, *MULTIPROCESSOR SYSTEM-ON-CHIPS*. Morgan Kaufamann Publisher, 2005.

[8] A. M. Amory, É. Cota, M. Lubaszewski, F. G. Moraes, **"**Reducing test time with processor reuse in network-on-chip based systems**,"** *Proceedings of the 17th ACM symposium on Integrated circuits and system design*, pp. 111 – 116, 2004.

[9] X., Jiang, W. Wolf, J. Hankel, S. Charkdhar, "A methodology for design, modeling and analysis for networks-on-Chip," *IEEE International Symposium on Circuits and Systems*, pp. 1778-1781 May 2005.

[10] A. Jantsch and H. Tenhunen. *NETWORKS ON CHIP* Kluwer Academic Publisher, 2003.

[11] Semiconductor Industry Association, The international Technology Roadmap for Semiconductors. 2001. http://public.itrs.net/Files/2001ITRS/Home.htm

[12] Y. Xiong, "An extensible type system for component-based design," Ph.D. dissertation University of California Berkeley, May 2002.

[13] E. Cota, M. Kreutz, C.A. Zeferino, L. Carro, M. Lubaszewski, A. Susin, "The impact of NoC reuse on the testing of core-based systems**,"** *21st Proceedings of VLSI Test Symposium*, pp. 128-133, April 2003.

[14] D. Bertozzi and L. Benini, "Xpipes: A network-on-chip architecture for gigascale system-on-chip," *IEEE Circuits and Systems*., vol. 4, no.1 pp. 18-31, 2004

[15] K. K.l Ryu, E. Shin, V. J. Mooney, "A comparison of five different multiprocessor SoC bus architectures," IEEE Euromicro symposium on Digital Systems, Design, pp. 202-209, 2001.

[16] J. Duato, S. Yalamanchili and L. Ni, M. "Interconnection Networks: An Engineering Approach," 1997.

[17] J. Duato "A New Theory of Deadlock-Free Adaptive Routing in Wormhole Networks," *IEEE Trans. on Parallel and Distributed Systems* vol. 4, pp. 1320-1331, 1993.

[18] E. Fleury and P. Fraigniaud "A General Theory for Deadlock Avoidance in Wormhole-Routed Networks," *IEEE Trans. on Parallel and Distributed Systems* vol. 9, pp. 626-638, 1998.

[19] L. Ni, M. and P. McKinley, K. "A Survey of Wormhole Routing Techniques in Directed Networks," *Computer* vol. 26, pp. 62-76, 1993.

[20] L. Schwiebert "Deadlock-Free Oblivious Wormhole Routing With Cyclic Dependencies," *IEEE Trans. on Computers* vol. 50, no. 9, pp. 865-876, 2001.

[21] L. Zakrevski "PhD Thesis: Fault-Tolerant Wormhole Message Routiing in Computer Communication Networks," *Boston University College of Engineering* pp. 21-27, 2000.

[22] - L. Zakrevski, S. Jaiswal, L. Levitin and M. Karpovsky "A New Method for Deadlock Elimination in Computer Networks with Irregular Toplologies," *Pro. of the IASTED Conf. PDCS-99* vol. 1, pp. 396-402, 1999.

[23] L. Zakrevski and M. Karpovsky, G. "Fault-Tolerant Message Routing in Computer Networks," *Proc. of Int. Conf. on PDPA-99* pp. 2279-2287, 1999.

[24] L. Zakrevski, S. Jaiswal and M. Karpovsky "Unicast Message Routing in Communication Networks With Irregular Topologies," *Proc. of CAD-99, 1999.*

[25] L. Zakrevski, M. Mustafa and M. Karpovsky "Turn Prohibition Based Routing in Irregular Computer Networks," *Proc. of the IASTED International Conference on Parallel and Distributed Computing and Systems* pp. 175-179, 2000.

[26] - D. Starobinski, M. Karpovsky and L. Zakrevski "Application of Network Calculus to General Topologies Using Turn Prohibition," *IEEE/ACM Transactions on Networking* vol. 11, no. 3, pp. 411-421, 2003

[27] M. Mustafa, M. Karpovsky and L. Levitin "Cycle Breaking in Wormhole Routed Computer Communication Networks," *OpnetWork2005, 2005*