Online NoC Switch Fault Detection and Diagnosis Using a High Level Fault Model

Armin Alaghi, Naghmeh Karimi, Mahshid Sedghi, Zainalabedin Navabi
Electrical and Computer Engineering Department
Faculty of Engineering – Campus #2 – University of Tehran, 14399 Tehran, IRAN
{armin, naghmeh, mahshid, navabi}@cad.ece.ut.ac.ir

Abstract

This paper presents an efficient method for online testing of NoC switches. This method deals with control faults of NoC switches; i.e. the routing faults which cause NoC packets to be sent to output ports not intended to. A high level fault model has been proposed in this paper to model switch routing faults. The proposed method is evaluated by fault simulation that is based on our high-level fault model. This simulation and evaluation environment is modeled at the transaction level in VHDL.

1. Introduction

With the growth of VLSI technology and moving toward nano scale domain, system-on-chip designs will contain billions of transistors in a single chip. However in an SoC with a large number of memory cells and hundreds of IP cores, traditional bus structures cannot handle the high volume of communication between cores. Utilizing global buses in an SoC implies high power consumption, unpredictable delay as well as synchronization errors. Thus there is a trend toward designing a scalable, reusable and predictable SoC-based architecture [1].One such emerging approach is the Network on Chip (NoC) Architecture [2].

NoCs are packet switch networks implemented on a single chip providing high performance interconnection to embedded cores [3]. A typical NoC consists of three main parts: switches to route the data packets, interfaces that connect each core to a switch in a NoC, and interconnections among the switches [4].

NoC design is a three dimensional design space. The first dimension deals with interconnection of the network nodes to each other (network topology), while the second dimension defines the form of message passing among the NoC nodes (routing mechanism). Finally, the third dimension represents the mapping of an application to the NoC nodes (application mapping) [2].

To design a NoC structure various tradeoffs regarding throughput, latency, silicon area, power consumption and reliability should be considered [5].

The failure rate in emerging nano technologies is estimated to be in order of 10-2 to 10-1 due to the shrunk size and frequency characteristics of these technologies in comparison with 10-9 to 10-7 failure rate in CMOS technologies [6, 7]. This unreliability of nano and deep submicron technologies makes online testing techniques essential.

In this paper we focus on online error detection and diagnosis in NoC switches. The rest of this paper is organized as follows. First a review of related works and backgrounds is given in Section 2 and 3 respectively. Section 4 describes general NoC faults. The proposed switch fault model is presented in Section 5. Our self testable switch is discussed in Section 6. Then experimental results of applying the proposed scheme to a number of NoC structures is given in Section 7 followed by the conclusion remarks in Section 8.

1550-5774/07 $25.00 © 2007 IEEE
DOI 10.1109/DFT.2007.55

2. Related works

The problem of error detection and diagnosis in NoC architectures has been studied thoroughly in the scope of offline testing.

Compared to traditional SoCs, test of NoC-based architectures involves multiple challenges due to the existence of complex network components [8]. Different research groups have proposed the reuse of the communication infrastructures as a Test Access Mechanism (TAM) [8, 9].

Nahvi et al. propose the use of a packet switch communication-based TAM, so called NIMA, for an SoC. Since NIMA has been primarily designed for testing, routing and addressing strategies are defined considering only the test requirements of each system [10].

Aktouf [11] suggests the use of a boundary scan wrapper to test NoC components. The test includes the routers, the RAM blocks, and the embedded processors of the NoC architectures.

A test strategy for NoC routers based on partial scan and on-chip response evaluation has been proposed in references [12] and [13].

Ubar [14] deals with BIST strategies to test NoC architectures.

Another BIST method for testing inter-switch links in a NoC structure has been proposed in reference [15]. This method uses a high level fault model [16] to deal with the crosstalk effects due to inter-wire coupling.

Another test methodology for testing the NoC switches has been presented in references 4 and 17. This methodology broadcasts same set of test vectors to all switches of a NoC structure and detects existing faults through comparison of switches outputs with each other.

With transient and intermittent faults becoming a dominant failure mode in modern VLSI, widespread deployment of online test approaches has become crucial. Traditionally, error detection and correction mechanisms are used to protect communication subsystems against the effects of transient malfunctions. What follows discusses some of the previous works in online testing.

Murali et al. [18] consider error detection codes to build self checking NoCs. They classify error detection codes for NoC applications to switch-to-switch (s2s) and end-to-end (e2e) categories. In the former category all the switches located in the path of the packet sender node and the packet receiver node check the packet to detect the probable faults while in the latter case just the sender and receiver nodes deal with the validity of the packet.

For self checking circuits two frequently used error detection codes are parity checking and dual rail codes. Code disjoint switches along with parity checkers can also be used for online fault detection and diagnosis in NoC communication fabrics [19]. Comparing the method presented in reference 19 with e2e and s2s approaches [18] shows the effectiveness of using code disjoint schemes over e2e and s2s approaches in terms of latency, power consumption and throughput.

A number of approaches to achieve fault tolerant NoC architectures have also been presented in the literature [22, 23]. Reference 22 discusses various types of reliability hazards in NoC structures and proposes a number of recovery techniques for reliability enhancement in presence of reliability hazards.

In general, the choice of an error recovery technique for an application requires considering different parameters, i.e., power, performance, and reliability tradeoffs [18].

Kim et al. [23] classify soft errors that disturb the correct operation of the NoCs as link and router errors. The former occurs during the traverse of flits from one router to another while the latter occurs within the router architecture. Considering separate error coding techniques for header flit, five types of retransmission techniques are used to remove link errors in this approach. Moreover a number of transient fault protection techniques are used in this method to deal with router errors.

Along with testing techniques for synchronous NoCs, test of asynchronous NoCs that are used for Globally Asynchronous Locally Synchronous (GALS) platforms has also been studied in the literature and a number of test architectures have been proposed [20, 21].

3. Preliminaries

As discussed in the previous sections, an NoC architecture is composed of a set of structured switches and point to point channels interconnecting the processing cores of a SoC in order to support communication among them [24].

In this paper we focus on regular 2-D NoC topologies using XY routing algorithm where a packet is first routed in the X direction and then in the Y direction before reaching the destination. Switching is based on wormhole approach where a packet consists of multiple fixed length control flow units (flits).

Figure 1 shows the structure of the NoC architecture used in this paper. As shown in this figure each switch has five I/O ports, 4 ports for connecting each switch to adjacent switches and 1 port to connect the switch to its related processor.

4. NoC faults

From the discussion in the previous sections we can infer that soft errors that affect the NoC structures are classified as link and router errors. Link errors can be detected via error coding techniques such as CRC and dual rail codes [19].

The impact of soft errors on data flits is temporary and the data is not actually corrupted. However, soft errors can cause erroneous behavior in the routing process. Because of this we primarily focus on control faults causing control errors. A control fault causes a packet be sent to a wrong output port of a switch.

To handle control faults we propose a high level fault model along with a detection and diagnosis methodology in the sections that follow. Figure 2 illustrates the switch architecture in more details.

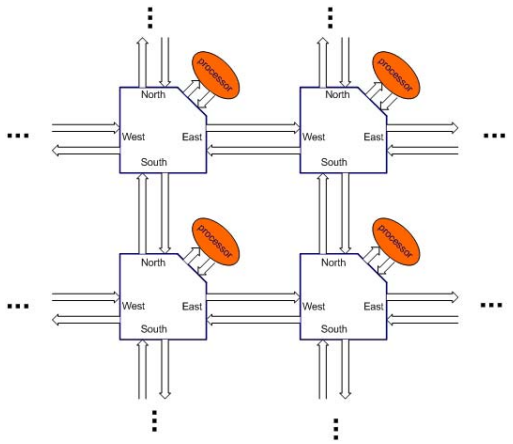

Figure 1. A regular 2-D mesh based NoC

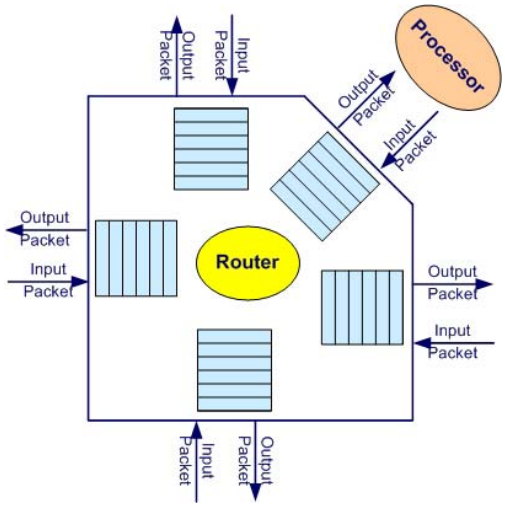

Figure 2. NoC switch structure

5. Proposed fault model

As discussed in Section 4, control faults in an NoC switch architecture cause switch packets to be sent to wrong ports.

One of the possible consequences of such an incorrect routing can be that all packets in an input port of a switch are sent to a specific output port; we call such a fault "stuck at port fault". Accordingly, 5 stuck-at direction faults can be identified:

1. Stuck-at East, SaE
2. Stuck-at South, SaS
3. Stuck-at West, SaW
4. Stuck-at North, SaN
5. Stuck-at Processor, SaP

When a Stuck-at East fault occurs all packets are sent to the East output port (Port 1) of the switch regardless of their destination fields. Similarly, other faults cause packets to be sent to their stuck-at ports. A stuck-at processor fault in a switch makes all the packets be sent to the related processor.

Other switch control faults such as packet dropping, lost-destination and misrouting can also be modeled with our suggested set of faults. However, these faults are beyond the scope of this paper. For example, consider the case when switch i corrupts the destination field of packets that it receives and sets this field to switch j. This fault in switch i can be modeled with a SaP fault in switch j.

6. Self testable switches

The function of a switch is sending a message from its input ports to its output ports according to the routing information stored in the header of that message. Routing unit consists of datapath and controller parts. The datapath deals with message transportation circuitry and the control part considers the hardware that decides which port the data must be routed to [19].

As discussed in the previous sections, due to the unfavorable impact of soft faults on control part, in this paper we consider control faults and model them by a high level fault model. Several methods of fault detection and their pros and cons are described in Sections 6.1 to 6.3.Wherever Times New Roman is specified, Times Roman, or Times may be used. If neither is available on your word processor, please use the font closest in appearance to Times New Roman that you have access to. Please avoid using bit-mapped fonts if possible. True-Type 1 fonts are preferred.

6.1. Distraction detection method

Our online fault detection and diagnosis, called *distraction detection*, is applied to meshed-based NoCs with XY routing algorithms. In this method, first switches extract routing information of the receiving packets and following this, based on the XY routing algorithm it is decided weather the packet was supposed to pass through this switch or it has been distracted.

The hardware implementation of *distraction detection* method is shown in Fig. 3. As shown , an extra hardware has been added to each switch to perform online fault detection and diagnosis. Considering the fact that in XY routing algorithm, a packet is first routed on X direction and then on Y direction, either the Y location of the switches transmitting a packet from a source node to a destination node should be equal to the Y

location of the source node, or the X location of the transmitting switches should be equal to the X location of the destination node. Thus the added hardware to each switch performs the comparisons between source and destination of each packet with the switch location. These comparisons are performed during the normal operation of the NoC, i.e., in this NoC structure a control fault is detected as soon as the fault occurs.

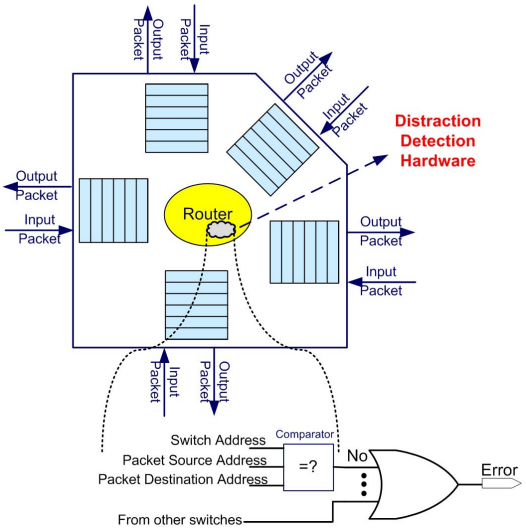

Figure 3. Self testable switch

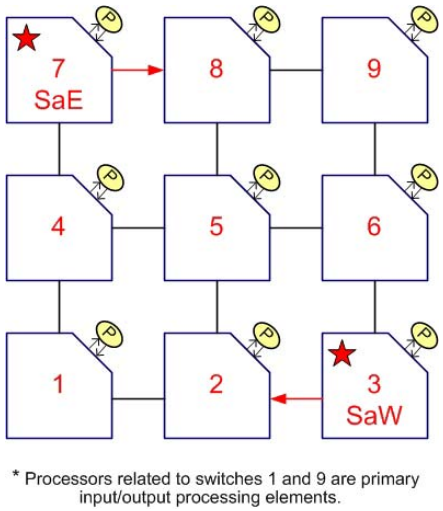

Figure 4. A sample 3*3 NoC architecture

In the 2-D mesh-based architectures the *distraction detection* method still leaves a few control faults undetected. These faults cause a packet to be transferred between two neighboring switches forever. This packet might still be in the routing path and so the corresponding fault cannot be detected by the distraction detection method.

For example consider a 3x3 NoC architecture shown in Fig. 4. To clarify the proposed algorithms, an ID number has been aassigned to each switch. Assume that only the processors related to switches 1 and 9 are externally accessible (the related processors act as primary input/output processing element). In this figure, the SaE fault of switch 7 and SaW fault of switch 3 cannot be detected by *distraction detection* method.

6.2. Switch count method

To enhance online fault detection in mesh-based NoCs, *switch count* approach is proposed. In this method each packet includes a switch count field which is incremented automatically by 1 when the packet passes through a switch. The overflow of this field is the evidence of a fault in the NoC structure. For example, assume that a packet from switch 1 intends to go to switch 9 in Fig. 4. In this case if switch 3 is SaW, the packet is sent to switch 2 after it is received by switch 3, and it sent back and forth between switch 2 and 3 several times.

Since both of the Y locations of switch 2 and 3 are equal to the Y location of the source switch (switch 1), the SaW fault of switch 3 is not detected by *distraction detection* method. However, using *switch count* approach, the switch count field of the transmitting packet overflows due to the several rotation of the packet between switch 2 and 3. Applying distraction detection method along with the switch count method, all stuck at port faults of all NoC switches except SaP faults can be detected.

6.3. Trapped Packet Detection Method

To detect the SaP faults, a small additional hardware has been inserted in each processor. This extra hardware compares the destination field of each incoming packet to the processor with the XY location of the related switch. A SaP fault is detected if a mismatch occurs. This method is called *trapped packet detection* since the packet is trapped in the processor and has no way out.

6.4. Fault Diagnosis Method

The next task after online fault detection in NoC structures is faulty unit identification. Unlike faults detected by the *distraction detection* method, the faults detected by the *switch count* method cannot be diagnosed, i.e., the exact location of these faults cannot be determined.

To diagnose a faulty switch by applying the *distraction detection* method, first each switch checks the source and destination fields of the received packet to detect the possible fault as discussed above. Then if a fault is detected, the switch reports its neighboring switch (the switch from which the packet is received) as faulty and sends a packet to the primary output switch(s). This packet reports the exact location of the faulty switch to the primary output port. After the fault diagnosis, proper actions can be taken to recover from the occurred fault. Applying recovery methods are beyond the scope of this paper.

The SaP faults detected by the processors cannot be diagnosed. In this case when a faulty switch receives a packet, forwards it to the related processor. This processor detects the related SaP fault but cannot report the fault location. This is because if the processor generates a packet to show the fault location and send the packet to the related switch, the generated packet will be forwarded to the processor itself and cannot be sent out of the switch due to of SaP fault in that switch.

Note that the proposed methods for online fault detection support multiple faults as well as single faults. However, the discussed diagnosis scheme is just applied to single fault models.

7. Experimental results

The proposed online test strategy was evaluated for three different NoC sizes, 3x3, 5x5 and 7x7, with respectively 9, 25 and 49 processors and switches assuming various packet traffics. Our on-line testing and diagnosis methods were simulated in a high level VHDL based platform [25].

In the selected NoCs, random packets were routed through the NoCs and the fault coverage (the number of detected control faults over the number of all injected control faults) was reported.

The results of applying our first online test method (*distraction detection* method) to the discussed NoCs are given in Table 1. In this table "addressed switch" is the number of switches that have been addressed as the destination for random generated packets over the total number of switches, i.e., in the 100% addressed switch scenario, all NoC switches become the destination of at least one random generated packet. Table 2 shows the resulting fault coverage applying *distraction detection* method along with the *switch count* method to the test case NoCs.

Table 1. Distraction detection

Addressed Switch / NoC Architecture	25%	50%	75%	100%
3x3	30	45	67	67
5x5	36	57	64	74
7x7	32	63	71	76

Table 2. Distraction detection with switch count

Addressed Switch / NoC Architecture	25%	50%	75%	100%
3x3	39	51	73	73
5x5	44	65	69	76
7x7	52	75	77	77

As discussed earlier, the SaP faults should be detected by the related processing elements. Tables 3 and 4 show the results of applying *trapped packet* detection along with previous discussed methods.

Table 3. Distraction detection with trapped packet detection

Addressed Switch / NoC Architecture	25%	50%	75%	100%
3x3	45	67	94	94
5x5	52	79	88	98
7x7	48	84	93	98

Table 4. Distraction detection with switch count and trapped packet detection

Addressed Switch / NoC Architecture	25%	50%	75%	100%
3x3	58	73	100	100
5x5	60	91	95	100
7x7	63	94	100	100

To evaluate the hardware overhead imposed by applying our methods, we applied the proposed methods to an existing NoC switch [4] and synthesized the resulted online-testable structures. Table 5 shows the area overhead imposed by applying the proposed methods to the existing NoC switch. The area overhead for all NoC sizes was approximately the same.

Figure 5 shows a comparison between these methods. As shown, the fault coverage resulted by applying only *distraction detection* method and the combination of *distraction detection* and *switch count* methods are very close to each other. However, the hardware overhead of applying *distraction detection* method is much lower than the combined method. The same observation exists when comparing the 3rd (*distraction detection* combined with trapped packet detection) and the 4th (*distraction detection* combined with *switch count* and *trapped packet detection*) methods.

Table 5. Area overhead percentage of applying proposed methods to an existing NoC

Area Overhead / Methods*	Switch Overhead (%)	Procor Overhead (%)	Total Overhead (%)
Method 1	1.6	0	0.2
Method 2	9.3	0	1.4
Method 3	1.6	0.3	0.5
Method 4	9.3	0.3	1.6

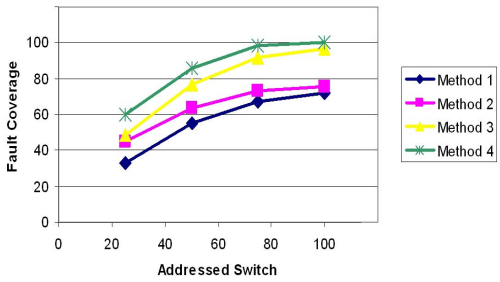

Figure 5. Comparison of proposed online fault detection method

*Method 1: Distraction Detection
Method 2: Switch Count + Distraction Detection
Method 3: Trapped Packet + Distraction Detection
Method 4: Trapped Packet + Distraction Detection + Switch Count

As discussed in the previous section when a control fault is detected using distraction detection method, a packet is sent to the primary output processing element(s) of the NoC structure to report the exact location of the faulty switch. Thus the results reported in Table 1 are also valid for fault diagnosis using *distraction detection* method. However, as discussed in Section 6, the exact location of control faults cannot be diagnosed by applying other proposed methods.

The area overhead percentage of our *distraction detection* diagnosis method was about 4.2% while the Switch speed reduced by 13.8%.

8. Conclusion

In this paper, we proposed an efficient scheme for online control fault detection and diagnosis in NoC switches. To model these faults a high level fault model is proposed. Synthesis and fault simulation results show that by accepting a low area overhead, the NoC switches become self testable.

9. References

[1] U. Y. Orgas, and R. Marculescu, Communication-Based Design for Nanoscale Socs, VLSI Handbook, Wai-Kai Chen(ed.), 2nd Edition, CRC Book Press, 2006, Chapter 16.

[2] L. Benini, and G. De Micheli, "Networks on Chips:A Paradigm," IEEE Trans. on Computer, Vol. 35, Jan 2002, pp. 70-78.

[3] C. Liu, V. Iyengar, J. Shi, and E. Cota, "Power-Aware Test Scheduling in Network-on-Chip Using Variable-Rate On-Chip Clocking," Proc. VTS, 2005, pp. 349-354.

[4] M. Hosseinabadi, A. Dalirsani, and Z. Navabi, "Using the Inter- and Intra-Switch Regularity in NoC Switch Testing," Proc. Design Automation and Test in Europe(DATE), 2007, pp. 361-366.

[5] P. P. Pande, G. De Micheli, C. Grecu, A. Ivanov, and R. Saleh, "Design, Synthesis, and Test of Networks on Chips," IEEE Trans. on Design and Test of Computers, Vol. 22, No. 5, Sep.-Oct. 2005, pp. 404-413.

[6] European Commission, Technology Roadmap for Nanoelectronics, 2001.

[7] K. Nikolic, A. Sadek, and M. Forshaw, "Architectures for Reliable Computing with Unreliable Nanodevices," Proc. IEEE NANO, 2001, pp. 254.259.

[8] B. Vermeulen, J. Dielissen, K. Goossens, and C. Ciordas, "Bringing Communication Networks On-Chip: The Test and Verification Implications," IEEE Communications Magazine, Vol. 41, No. 9, Sept. 2003, pp. 74-81.

[9] E. Cota et al., "Power aware NoC Reuse on the Testing of Core-Based Systems," Proc. International Test Conference (ITC) 2003, pp. 612-621.

[10] M. Nahvi, and A. Ivanov, "Indirect Test Architecture for SoC Testing," IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Vol. 23, No. 7, 2004, pp. 1128-1142.

[11] C. Aktouf, "A Complete Strategy for Testing an on-chip Multiprocessor Architecture," IEEE Trans. on Design and Test of Computers, Vol. 19-1, 2002, pp. 18-28.

[12] A. M. Amory, E. Briao, E. Cota1, M. Lubaszewski, and F. G. Moraes, "A Scalable Test Strategy for Network-on-Chip Routers," Proc. International Test Conference (ITC), 2005.

[13] A. M. Amory, E. W. Brião1, É. F. Cota1, M. S. Lubaszewski, , and F. G. Moraes, "A Cost-Effective Test Flow for Homogeneous Network-on-Chip,"Proc. European Test Symposium (ETS), 2005.

[14] R. Ubar, and J. Raik, Testing Strategies for Network on Chip, Networks on Chip, A. Jantsch and H. Tenhunen (ed.), Kluwer Academic Publisher, 2003, pp. 131-152.

[15] C. Grecu, P. Pande, A. Ivanov, and R. Saleh, "BIST for Network-on-Chip interconnect infrastructures," Proc. 24th IEEE VLSI Test Symposium (VTS'06), 2006, pp. 30-35.

[16] M. Cuviello, S. Dey, X. Bai, and Y. Zhao, "Fault Modeling and Simulation for Crosstalk in System-on-Chip Interconnects," Proc. IEEE/ACM International Conference on Computer-Aided Design, 1999, pp. 297-303.

[17] M. Hosseinabadi, A. Banaiyan, M. N. Bojnordi, and Z. Navabi, "A Concurrent Testing Method for NoC Switches," Proc. Design Automation and Test in Europe(DATE), 2006, pp. 1171-1176.

[18] S. Murali, G. De Micheli, L. Benini, T. Theocharides, N. Vijaykrishnan, and M. Irwin, "Analysis of Error Recovery Schemes for Networks on Chips," IEEE Trans. on Design and Test of Computers, Vol. 22, No. 5, 2005, pp. 434-442.

[19] C. Grecu1, A. Ivanov, R. Saleh, E. S. Sogomonyan, and P. P. Pande, "On-line Fault Detection and Location for NoC Interconnects," Proc. 12th IEEE International Symposium on On-Line Testing (IOLTS), 2006, pp. 145-150.

[20] X. T. Tran, J. Durupt, F. Bertrand, V. Beroulle, and C. Robach, "A DFT Architecture for Asynchronous Networks on-Chip," Proc. 11th IEEE European Test Symposium (ETS'06), 2006, pp. 219-224.

[21] E. Beigne, F. Clermidy, P. Vivet, A. Clouard, and M. Renaudin, "An Asynchronous NoC Architecture Providing Low Latency Service and Its Multi-Level Design Framework," Proc. 11th International Symposium on Asynchronous Circuits and Systems (ASYNC), 2005, pp. 54–63.

[22] D. Park, C. Nicopoulos, J. Kim, N. Vijaykrishnan, and C. R. Das, "Exploring Fault-Tolerant Network-on-Chip Architectures," Proc. International Conference on Dependable Systems and Networks (DSN'06), 2006, pp. 93-104.

[23] J. Kim, D. Park, C. Nicopoulos, N. Vijaykrishnan, and C. R. Das, "Design and Analysis of an NoC Architecture from Performance, Reliability and Energy Perspective," Proc. Symposium on Architecture for Networking and Communications Systems (ANCS), 2005, pp. 173-182.

[24] E. Cota, C. Zeferino, M. Kreutz, L. Carro, M. Lubaszewski, and A. Susin, "The Impact of NoC Reuse on the Testing of Core-based Systems," Proc. IEEE VLSI Test Symposium, 2003, pp.128-133.

[25] M. Sedghi , A. Alaghi, E. Koopahi, and Z. Navabi, "An HDL-Based Platform for High Level NoC Switch Testing,", submitted to Asian Test Symposium (ATS), 2007.