Reliable NoC Architecture Utilizing a Robust Rerouting Algorithm

[1]Armin Alaghi, [2,1]Mahshid Sedghi, [1]Naghmeh Karimi, [2]Mahmood Fathy, [1]Zainalabedin Navabi
[1]Electrical and Computer Engineering Department, University of Tehran
[2]Computer Engineering Department Iran University of Science and Technology
a.alaghi@ece.ut.ac.ir, ma_sedghi@comp.iust.ac.ir, naghmeh@cad.ece.ut.ac.ir,
mahfathy@iust.ac.ir, navabi@cad.ece.ut.ac.ir

Abstract

Moving towards reconfigurability is an approach to increase fault tolerance on System-on-Chip design. In this paper, we propose a self-reconfigurable NoC architecture utilizing a robust rerouting method. At first, an offline test strategy for locating system-level faults in NoC switch ports is utilized. Using the information achieved in the test phase, every switch reconfigures itself to avoid routing packets through faulty links by utilizing our local rerouting method. The proposed rerouting method is evaluated using a Transaction-Level platform. Experimental results show that our proposed rerouting method delivers all the packets in a faulty NoC successfully and has a less communication overhead compared to a pure flooding method.

1 Introduction

Over the past few years, Network-on-Chip (NoC) has become increasingly popular as a scalable interconnect infrastructure for IP cores. A NoC replaces the slow ad-hoc global on-chip wiring with a high performance communication infrastructure which facilitates structured modular system design and thus helps reducing the system design complexity. NoCs are characterized by different tradeoffs regarding throughput, latency, silicon area, power consumption and reliability [1][2].

Moving towards nano-scale circuits poses new challenges to design of digital circuits. Shrinking dimensions result to a significant decrease in the manufacturing yield. Though, the yield can be maintained at an acceptable level by allowing some amount of faults occur in the chip and utilize fault tolerance techniques to provide reliable functionality. Reconfigurability is one of the approaches to increase fault tolerance in SoC design [9].

Electronic System-Level Design (ESL) has gained popularity in design of digital circuits recently. Transaction-Level Modeling (TLM) is going to become the starting point in ESL. TLM improves simulation performance and modeling efficiency for early design space exploration. SystemC has been

proved to be the key to the fairly fast deployment of this methodology. In a general definition of TLM, the system is divided into two parts: communication and computation parts. TLM models communication parts of a system at a high level of abstraction (e.g., by functions) [11].

In this paper, we introduce a reliable mesh-based NoC architecture utilizing a robust rerouting method. An offline test strategy is proposed which locates system-level faults in switch ports. Using the information achieved in the test phase, every switch reconfigures itself to avoid routing packets through faulty links by utilizing a local rerouting algorithm. A Transaction-Level platform is used for simulation and evaluation of our test strategy and rerouting technique. Comparing to a VHDL platform, TLM provides us an appropriate environment for fast simulation of the proposed techniques.

The rest of this paper is organized as follows. Section 2 reviews related works briefly. The method for locating faults in switch ports is explained in Section 3. Section 4 deals with the rerouting method and self-reconfigurable switch architecture following by experimental results presented in Section 5. Finally, Section 6 concludes the paper.

2 Related Works

Kim et al. [8] classify errors disturbing 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. Amory at al. propose a partial scan method along with a test wrapper to test NoC routers [6]. In this scheme all routers have the same number of scan chains and are tested simultaneously with the same test data. A testing method for NoC FIFO buffers using a distributed BIST scheme is proposed in [7]. In this method the read/write mechanism, control circuit and test data are shared among the FIFO blocks while the response analyzers for each FIFO are distributed.

A BIST method for testing NoC links has been proposed in [5] which uses a high level fault model to deal with the crosstalk effects due to inter-wire coupling. Raik et al. [4] use different test

configurations to diagnose link faults. They assume that link faults result in dropped or corrupt data passing through a link.

A wide variety of fault-tolerance techniques for the NoCs have been proposed in the literature. Most of the techniques consist of various forms of gossip algorithms [12]. In [13], a random walk algorithm is proposed which sends a finite, predetermined number of copies of a message into the network. This algorithm result in a significantly less communication overhead compared to pure *flooding*. In [3], a number of architectural techniques to prevent or recover from the impact of transient errors on NoC links are presented which use a new hop-by-hop retransmission scheme. A dynamic routing mechanism for the NoCs is proposed in [14] which is a simplified form of *Link State* routing. This method promises successful communication in case of both link and router faults.

A number of groups utilize reconfigurability techniques in order to make the NoC fault-tolerant. Honarmand et al. propose a heuristic method for reconfiguring NoC switches in the presence of faults in links or switches [10]. They extract a number of constraints that should be met to have a live-lock free routing. In [9], a switch node architecture with automatic rerouting property is proposed. The rerouting algorithm can be used in order to avoid faulty or congesting ports.

From the hardware description point of view, TLM (Transaction Level Modeling) has gained wide spread acceptance in the system-level design community. TLMs are applied on different abstraction levels and for very different purposes [15]. In [16], an asynchronous NoC protocol is proposed and implemented using TLM. Some research groups utilize TLM simulation platforms for performance/power evaluation of different NoC architectures [17][18]. In [19], a reliable NoC architecture is proposed and evaluated using a TLM platform. The architecture used in this work is an application-specific mapped mesh-based NoC.

In this paper, we use a TLM platform for evaluating our proposed test strategy and reconfiguration techniques. To the best of our knowledge, using TLM for testing NoC architectures is not yet addressed by the research community.

3 Port Fault Diagnosis Method

In this paper, we use a high-level fault model which is based on the functionality of ports. A fault-free port transfers data correctly, while a faulty port drops the packet.

Our TLM NoC model is a 2-D mesh of switches defined as SystemC modules communicating with each other through channels of type *tlm_fifo*. Each switch

has five pairs of input/output ports; four of them connect the switch to its neighboring switches in the mesh and one connects the switch to its corresponding processor. The inputs and outputs of our NoC are placed at PI (the switch at bottom-left of the NoC) and PO (the switch at bottom-right) nodes respectively.

Each switch keeps the status of all of its ports. All ports are suspected to be faulty at the beginning. At the start of the test session, the NoC PI generates a test packet and sends copies of the packet to all of its neighbors. This is called *flooding*. Upon receiving a test packet, each switch sends an acknowledgement (*ack*) to the port from which the test packet arrives and floods it. A copy of the packet is also sent to the processing element connected to the switch. Only when an *ack* is received from an input port, the switch will know that the specific input port is fault-free. The test session ends when there are no further test or *ack* packets in travel throughout the network. At this time, every switch has a thorough knowledge of the status of its ports. The switch will use this information later for reconfiguration and rerouting purposes.

4 Self-Reconfiguring the NoC

After detecting and diagnosing the NoC link faults, each switch receives the information about its neighbors. This information is enough for the switches to reconfigure their own routers with a local rerouting algorithm. Our proposed routing algorithm is a parameterized self-reconfigurable algorithm based on simple local routing rules.

The routing rules can be illustrated through Figure 1.a to Figure 1.d. In these scenarios, a packet whose destination is located two switches above the illustrated switch arrives.

The first routing rule is that if a switch has only one fault-free port, all packets should be routed through that port. This rule can be seen Figure 1.a. The second rule is that a switch never routes a received packet to its incoming port, unless it has only one fault-free port available. In other words, the port a packet comes from in considered as a faulty port for the router like the situation shown in Figure 1.b. Figure 1.c illustrates another routing rule that says if a packet cannot be routed normally (to north in this case), route it randomly through an available port. The dotted arrows shown in Figure 1.c show that the packet is routed to east or west randomly. The final scenario shown in Figure 1.d is a fault-free switch that routes the received packet normally (like XY-routing algorithm).
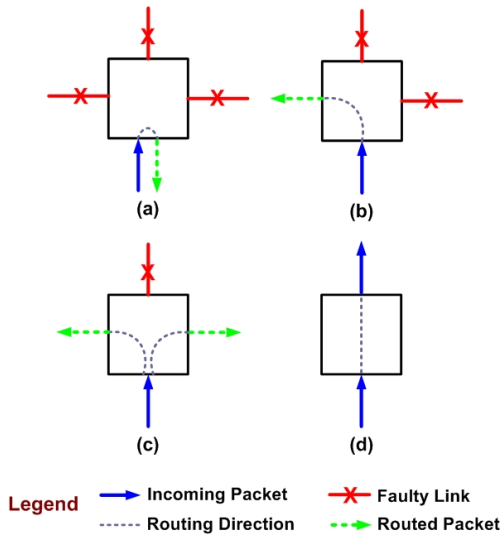
Figure 1. Proposed rerouting algorithm rules

Steps taken for a more complex routing than the above examples are illustrated in Figure 2. At step 0, a packet is generated at the switch marked as "Source" and is to be routed to the switch marked by "Destination". The switches marked by "X" are faulty switches. The first switch routes the packet to east normally (i.e., using the XY-routing algorithm). Switch 2 receives the packet at the end of step 1, and since it has only two healthy ports routes it to east at step 2.
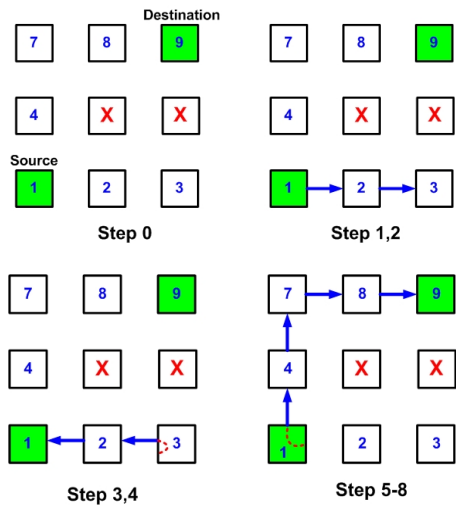

Figure 2. A more complex routing scenario

Switch 3 has only one available port and at step 3, routes the packet through that port (i.e., back to west). Switch 2 again receives the packet at step 4, but this time routes it to west according to the routing rule saying "a packet is never routed to its incoming port". At steps 5-8, the switches route the packet to its destination following the same rule.

The proposed routing algorithm is comparable to the XY-routing algorithm in terms of packet traffic since it does not multicast the packets. Moreover, experiments show that the proposed routing algorithm delivers approximately 95% the packets to their destinations successfully, which is far better than the XY-routing algorithm. In rare cases the packets fall into a deadlock or loop and our proposed algorithm fails to deliver the packets to their destinations. To make our algorithm robust, we added a flooding option to the switch routers. Each switch keeps the history of the received packets, and when it encounters a specific packet 5 times, it floods the packet instead of routing it normally. Adding this option to the switches successfully delivers all the packets of the NoC to their destination, at the cost of increasing the total NoC traffic. However, the traffic overhead can be ignored since the flooding happens in less than 5% of the situations.

5 Experimental Results

To evaluate our rerouting algorithm, we applied random multiple faults to our platform and generated packets to examine every possible path in the network. We also applied different routing algorithms, such as pure random and XY-routing algorithms to the NoC to compare them with our proposed algorithm. We examined NoCs with 9 switches up to 100 switches and for each NoC, we examined different fault probabilities for ports and ran numerous random experiments.
Figure 3 shows the percent of the usable switches of NoCs (z-axis) of different size (x-axis) with different fault probabilities (y-axis).
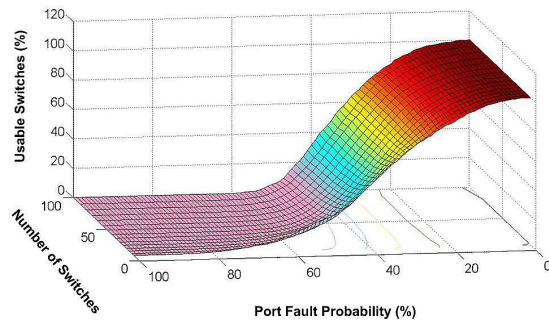

Figure 3. Total usable NoC switches

Figure 4 shows the number of usable switches when applying our proposed routing algorithms without the flooding option and
Figure 5 shows the same results for a simple XY-routing algorithm. As it can be observed, our proposed method uses nearly all of the usable switches where the XY-routing algorithm fails to use many available resources.
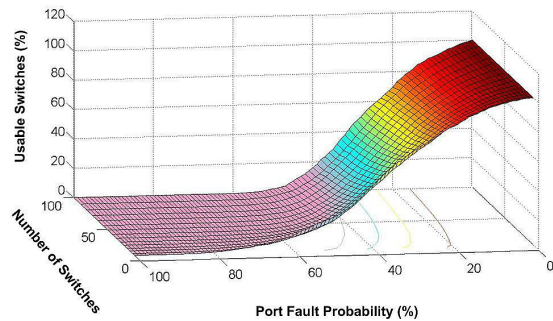
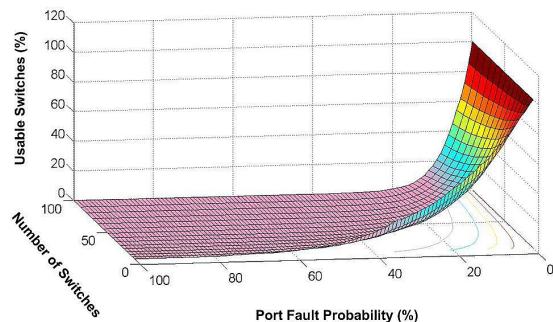**Figure 4. Usable NoC switches via the proposed routing algorithm**

Figure 5. Usable NoC switches via the XY routing algorithm

6 Conclusions

This paper introduced a reliable mesh-based NoC architecture utilizing a robust rerouting method. We proposed an offline strategy for diagnosing multiple faults in NoC switch ports. Using the information achieved in the test phase, every switch reconfigures itself to avoid routing packets through faulty links by utilizing a local rerouting algorithm. Experiments show that our proposed rerouting algorithm delivers almost all of the packets to their destinations successfully. To make our algorithm more reliable, we added a flooding option to the switch routers in order to save those packets that are trapped in deadlocks. This routing algorithm can also be used to dynamically avoid network congestions by temporally setting the congested links as faulty.

References

[1] N. K. Kavaldjiev, "A run-time reconfigurable Network-on-Chip for streaming DSP applications," Ph.D. thesis, University of Twente, 2006.

[2] 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, 2005, pp. 404-413.

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

[4] J. Raik, R. Ubar, and V. Govind, "Test Configurations for Diagnosing Faulty Links in NoC Switches," Proc. ETS, 2007, pp. 29 – 34.

[5] C. Grecu, P. Pande, A. Ivanov, and R. Saleh, "BIST for Network-on-Chip Interconnect Infrastructures," Proc. VTS, 2006, pp. 30-35.

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

[7] C. Grecu, P. Pande, B. Wang, A. Ivanov, and R. Saleh, "Methodologies and Algorithm for Testing Switch-Based NoC Interconnects," Proc. DFT, 2005, pp. 238-246.

[8] 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. Symp. on Architecture for Networking and Communications Systems (ANCS), 2005, pp. 173-182.

[9] P. Rantala, T. Lehtonen, J. Isoaho, and J. Plosila, "Fault-tolerant Routing Approach for Reconfigurable Networks-on-Chip," Proc. Int. Symp. on System-on-Chip, 2006, pp. 1-4.

[10] N. Honarmand, A. Shahabi, and Z. Navabi, "A Heuristic Search Algorithm for Re-routing of On-Chip Networks in The Presence of Faulty Links and Switches," Proc. IEEE EWDTS, 2007, pp. 411-416.

[11] F. Ghenassia (Ed.), Transaction-Level Modeling with SystemC: TLM Concepts and Applications for Embedded Systems, Springer, 2005.

[12] T. Dumitras, S. Kerner, and R. Marculescu," Towards onchip fault-tolerant communication," Proc. ASP-DAC, 2003, pp. 225-232.

[13] M. Pirretti, G.M. Link, R.R. Brooks, N. Vijaykrishnan, M. Kandemir, and M.J. Irwin, "Fault Tolerant Algorithms for Network-On-Chip Interconnect," Proc. ISVLSI, 2004, pp. 46-51.

[14] M. Ali, M. Welzl, and S. Hellebrand, "A Dynamic Routing Mechanism for Network on chip," Proc. NORCHIP Conf., 2005, pp. 70-73.

[15] T. Wild, A. Herkersdorf, and R. Ohlendorf, "Performance Evaluation for System-on-Chip Architectures using Trace-based Transaction Level Simulation," Proc. DATE, 2006, pp. 1-6.

[16] 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. IEEE Int. Symp. on Asynchronous Circuits and Systems (ASYNC), 2005, pp. 54-63.

[17] J. Xi, and P. Zhong, "A System-level Network-on-Chip Simulation Framework Integrated with Low-level Analytical Models," Proc. Int. Conf. on Computer Design (ICCD), 2006, pp. 383-388.

[18] H. Lebreton, and P. Vivet, "Power Modeling in SystemC at Transaction Level, Application to a DVFS Architecture," Proc. IEEE Computer Society Annual Symp. on VLSI, 2008, pp. 463-466.

[19] F. Refan, H. Alemzadeh, S. Safari, P. Prinetto, and Z. Navabi, "Reliability in Application Specific Mesh-Based NoC Architectures," Proc. IOLTS, 2008, pp. 207-212.