A High-Fault-Coverage Approach for the Test of Data, Control, and Handshake Interconnects in Mesh Networks-on-Chip

Érika Cota, *Member*, *IEEE*, Fernanda Lima Kastensmidt, *Member*, *IEEE*, Maico Cassel, Marcos Hervé, Pedro Almeida, Paulo Meirelles, Alexandre Amory, and Marcelo Lubaszewski, *Member*, *IEEE*

Abstract—A novel strategy for detecting interconnect faults between distinct channels in networks-on-chip is proposed. Short faults between distinct channels in the data, control, and communication handshake wires are considered in a cost-effective test sequence for mesh NoC topologies based on XY routing.

Index Terms—Fault coverage, interconnect testing, test generation, reliability.

1 Introduction

As the number of IP modules in Systems-on-Chip (SoCs) increases, bus-based interconnection architectures may prevent these systems from meeting the performance required by many applications. For systems with intensive parallel communication requirements, for example, buses may not provide the required bandwidth, latency, and power consumption [1]. A solution for such a communication bottleneck is the use of an embedded switching network, called Network-on-Chip (NoC), to interconnect the IP modules in SoCs [2].

Several works have addressed different aspects of NoC design and implementation [1], [2], [3]. Recently, industrial NoCs have appeared in the market [4]. However, to become an industrial reality, this new design paradigm still depends on the definition of feasible, efficient, and plug-and-play test mechanisms. Such mechanisms must tackle both the network itself and the IPs connected to the NoC. The reuse of the NoC as a Test Access Mechanism (TAM) has been proposed [4], [5], [6], [7] to test the embedded IPs in an NoC-based system. Thus, the test of the NoC itself becomes an even more important issue to ensure the SoC test quality.

The huge number of interconnects allied to the shrinking of the chip dimensions make the NoC prone to a growing number of interconnect faults. The capability of detecting interconnect faults in NoC-based SoCs such as short circuits between two channels is mandatory for yield improvement.

Recent works have been addressing the test of the NoC infrastructure, including routers [8], [9], [10], [11] and interconnect channels [11], [12], [13]. Interconnect testing in NoC-based chips has been related to faults in wires within a single channel connecting two adjacent routers. However, this assumption is not reasonable in large NoC layouts. Considering realistic NoC layouts [14], the placement and routing of routers and channels are actually prone to even simpler faults, such as shorts between wires connecting the core to the network and between wires of distinct network channels. Figs. 1a and 1b illustrate two possible floorplans for an NoC-based SoC. In both cases, as indicated in Fig. 1c, parts or the entire NoC are likely to be implemented in Standard Cells gathering different routers and channels in the same piece of layout. Interconnecting the Standard Cells makes use of different metal layers, as depicted in Fig. 1d. Thus, it is possible that short circuits or crosstalks involve interconnects implemented using metal lines belonging to the same or to different layers. For full-custom layout implementations, faults between wires of distinct channels are less likely, but can still be observed. So, it is mandatory to extend the fault model to include interaction faults that affect different channels of an NoC, considering at least a channel's neighborhood.

Additionally, it is interesting to keep the network in its functional mode to test the interconnections at speed. However, in this mode, a number of router ports and paths are not easily configurable or even feasible. Therefore, the main challenge is to devise a test strategy that copes with the NoC routing constraints while ensuring a high-quality efficient test. Moreover, faults involving control links should also be considered, in addition to data link faults.

To the best of our knowledge, the necessity of this extended model prevents the use of current NoC testing approaches since they deal with interswitch links only.

In this paper, a test strategy is proposed for the detection of shorts between pairs of wires (including data and control

- *É. Cota, F.L. Kastensmidt, M. Cassel, P. Almeida, P. Meirelles, and A. Amory are with the PPGC-Instituto de Informática, Universidade Federal do Rio Grande do Sul, PO Box 15064, ZIP 91501-970, Porto Alegre, RS, Brazil.*
  E-mail: {erika, fglima, mcassel, prvalmeida, prmmeirelles, amamory}@inf.ufrgs.br.
- *M. Hervé and M. Lubaszewski are with PGMICRO-Departamento de Engenharia Elétrica, Av. Osvaldo Aranha, 103, ZIP 90035-190, Porto Alegre, RS, Brazil. E-mail: marcos.barcellos@ufrgs.br, luba@ece.ufrgs.br.*

Manuscript received 2 July 2007; revised 15 Dec. 2007; accepted 13 Mar. 2008; published online 4 Apr. 2008.
Recommended for acceptance by R. Marculescu.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number TCSI-2007-07-0295.
Digital Object Identifier no. 10.1109/TC.2008.62.

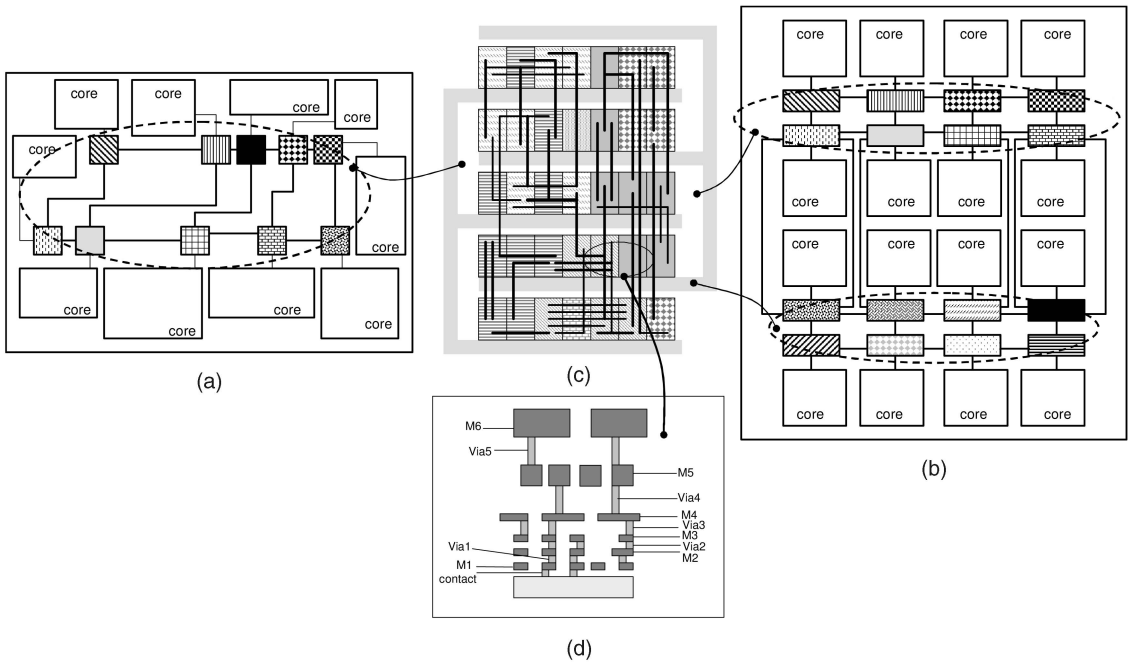

Fig. 1. (a) and (b) Floorplan examples for NoC-based systems-on-chip. (c) NoC routers implemented as standard cell cores. (d) Router interconnects across multiple metals/via levels.

wires within a single channel or between channels) for a mesh NoC with XY routing.

The contributions of this paper are threefold:

1. We show that interconnect faults can affect channels of nonadjacent routers in a NoC, including control wires and channels connecting the IP to the NoC, and this expanded fault model can prevent the application of current test strategies.
2. We propose a cost-effective test strategy that is capable of dealing with this expanded fault model for a 2×2 basic mesh NoC.
3. We propose a hierarchical strategy for testing the interconnects of larger mesh NoCs based on the strategy devised to cover any pairwise short faults in a 2×2 NoC neighborhood.

The well-known Walking-One Sequence is used in this work to exemplify our interconnect test solution. Furthermore, the proposed testing approach can be easily extended to other interaction faults in the connections, such as crosstalk, by simply adapting the test sequence.

This paper is organized as follows: Section 2 reviews recent works on NoC infrastructure test strategies. Section 3 explains the NoC and router topology and the corresponding fault model considered in this work. The proposed interconnect testing approach is presented in Section 4 and the overall test structure is detailed in Section 5. Section 6 contains the experimental results, while Section 7 explains how the method can be applied to larger NoCs. Section 8 concludes this paper.

2 Related Works

The test of an NoC-based SoC for manufacturing defects is usually divided into two parts: the test of the cores and the test of the communication infrastructure. The test of the cores is usually based on the reuse of the NoC as TAM [4], [5], [6], [7] to reduce the area overhead.

Works addressing the test of the NoC itself take advantage of the many identical (or very similar) structures, e.g., routers and network interfaces (NIs), to reduce the area overhead and/or accelerate test time. For instance, Amory et al. [8] propose a scalable methodology for testing NoC routers by using scan chains and a specific router configuration during test so that the same set of vectors can be broadcast to all routers while responses can be compared within the chip. Stewart and Tragoudas [9] proposed a functional fault model and a functional test strategy for routers and NIs of regular NoCs with a grid topology. Grecu et al. [10] propose two schemes for testing the combinational blocks of the NoC routers based on unicast and multicast transmission of test data packets through the switching fabric.

NoC interconnections are also regular, but present poor observability and controllability due to their density and central position. Thus, although all interconnects can use the same set of test vectors, ensuring its application to all wires is a challenge from the test time and fault-coverage point of view. Raik et al. [11] propose an external test method for NoCs based on functional fault models. The method targets single stuck-at faults at the multiplexers and registers inside the network routers, but the authors suggest that delay faults, opens, and shorts between wires within a single channel can also be covered.

Grecu et al. [12] propose a built-in self-test methodology for testing the channels of the communication platform. The proposed methodology targets crosstalk faults assuming the MAF fault model [16]. The authors also suggest that shorts between wires within a single channel are detected as well. The test strategy is based on two BIST blocks: the

test data generator (TDG) and the test error detector (TED). TDG generates the test vectors capable of detecting all possible crosstalk faults in a channel connecting two adjacent routers. The test vectors are launched on the channel under test from the transmitter side of the link and then are sampled and compared for logical consistency at the receiver side of the link by the TED circuit. Then, two approaches are discussed with respect to area overhead and test time. In the first approach, the interconnect testing is combined with the test of the routers [10]. Thus, a single TDG block is required to generate the MAF test patterns for all channels. The MAF test data is organized in test packets and a header is appended which identifies the channel that will be tested. When the test packet reaches the channel under test, the TED circuit compares incoming test vectors to the locally generated MAF data. The second approach uses multicast if this feature is available in the NoC under test. In this case, multiple packets can be sent over nonoverlapping channels such that more than one channel is tested at the same time.

In another approach, Pande et al. [13] propose to incorporate crosstalk avoidance coding and forward error correction schemes in the NoC data stream to enhance the system reliability.

All related works discussed so far do not consider interconnect faults involving different channels and the connections with the cores. This fact limits the efficacy of the entire testing. In order to observe this limitation, we have implemented the solution proposed in [12] to a basic 2×2 NoC (as detailed in Section 3) and injected all possible shorts (AND-short fault model) between any two wires of the NoC channels. In the first experiment, we injected faults between any 10 out of 12 wires of each channel, i.e., eight data wires and two control wires that signal the beginning of packet (bop) and end of packet (eop). The fault coverage for the extended fault model was 90 percent. The second experiment injected faults in all data, control, and handshake wires in the network. For this last experiment, the fault coverage dropped to 76.8 percent. Results showed the inadequacy of the approach in detecting faults between multiple channels.

Thus, one can observe that current approaches still do not tackle an extended fault model where shorts can happen beyond the limits of a single channel and protocol wires can also be faulty. In the next sections, we detail the extended fault model and propose a test strategy considering these faults. In this paper, we focus on the test of the NoC interconnections only.

3 NoC Case Study

We base our analysis on a 2D mesh topology packet-switching network model, the so-called SoC Interconnection Network (SoCIN), introduced in [3], [18]. SoCIN's router uses the Router Architecture for SoCs (RASoC). Fig. 2 depicts the RASoC router, which consists of five input and five output links. One pair of I/O links (called the local channel L) is dedicated to the connection between the router and the core; the remaining four pairs (north N, south S, west W, and east E) connect the RASoC router

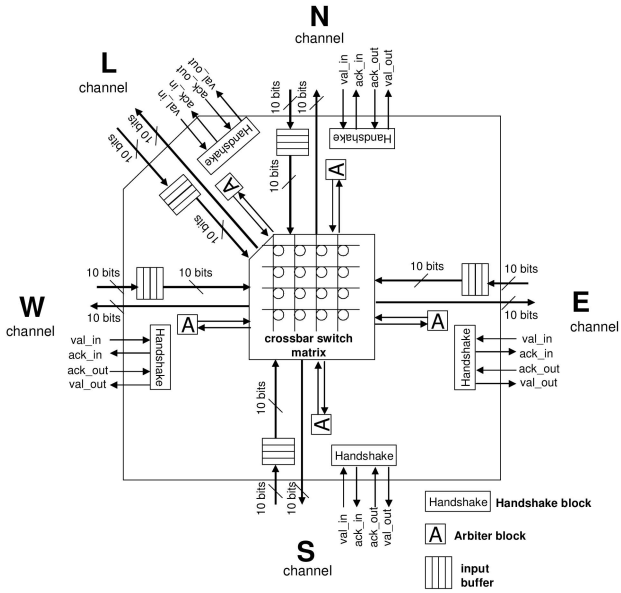

Fig. 2. RASoC router scheme.

with four adjacent routers. These channels include two unidirectional opposite links, each with data, framing (known as the bop and eop signals), and handshake communication signals.

The router is a VHDL soft core parameterized in three dimensions: communication link width, input buffer depth, and routing information width. The architecture uses the wormhole switching approach and a deterministic source-based routing algorithm. The wormhole approach breaks a packet into multiple flow-control units called flits. A flit is the smallest unit over which flow control is performed and its size is equal to the channel width. The network uses credit-based flow-control and XY routing, where a packet is first routed on the X direction and then on the Y direction before reaching its destination [17]. RASoC applies the handshake protocol for link flow control and uses round-robin arbitration and input buffering.

In all of the experiments presented here, we configured the RASoC communication channels with a four-position input buffer depth and the links with a 10-bit data width. Each communication link actually has 12 bits: eight bits for data, two extra wires for control (stored in the input buffer) called framing bits or the bop and eop bits, and two signals for the communication handshake between routers (val and ack). The entire case study system is composed of four routers and four nodes that are able to send and receive the packets containing the test vectors. We assume that, in each node, a core is connected to a router through an NI or wrapper.

3.1 Fault Space Definition

A short fault is a state in which a net is connected to another net, which can be placed at the same metal layer or at the top or bottom metal layers. There are three types of short faults: OR-short, AND-short, and strong driver. In the case of an NoC communication, this universe of faults can reach nets of many different interconnect channels, including router-to-router interconnect wires (rr_links) and router-to-core interconnect wires (rc_links) in both directions.

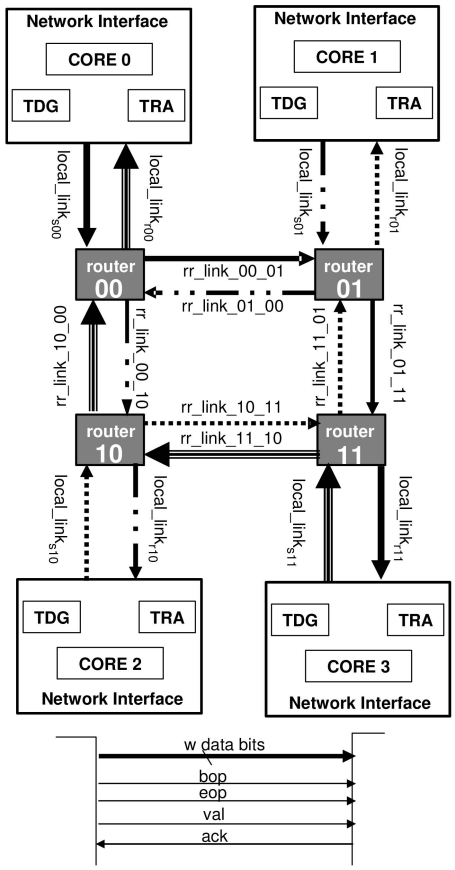

Fig. 3. Test strategy for the basic NoC topology.

When considering short faults in the NoC, it is important to define the region where the faults may occur, i.e., which links will more likely be short circuited. Some authors have been concerned about faults in router-to-router wires (rr_links) in a single direction inside a channel. On the one hand, the assumption that short faults will occur only between these unidirectional wires is too optimistic when dealing with a high-density design, as explained. On the other hand, considering that all possible wires can be faulty might not be realistic as well. The number of faults grows exponentially with the number of wires considered, as shown in (1) for n, the number of independent wires, and k, the size of each fault group, i.e., the number of wires that can be shorted together:

$$C(n,k) = \frac{n!}{k!(n-k)!}. \quad (1)$$

The affected region is actually defined during placement and routing steps. In some cases, faults can occur between any set of links of the network, so the searching region is the entire NoC. In other cases, faults may affect only the links placed in a specific part of the network, which reduces the search space. The challenge is thus to find a suitable trade-off in terms of testing time and fault coverage for the needed searching space.

In this work, we assume a faulty neighborhood composed of four routers (in a 2×2 grid configuration), four NIs that connect the cores to the network, and all wires (16 unidirectional links with data, control, and

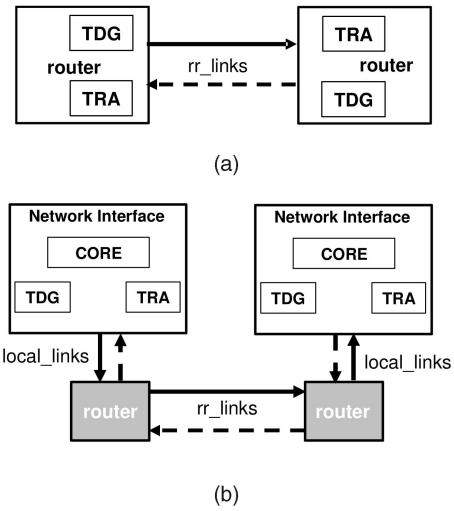

Fig. 4. BIST-based test technique. (a) BIST blocks embedded in the routers. (b) BIST blocks embedded in the network interface of cores.

communication handshake wires) connected to these components (routers and NIs), as illustrated in Fig. 3. This configuration was defined by taking the average connection lengths and the probability of shorts into different metal layers into account [25]. In the worst-case scenario, shorts can occur between any two wires of this entire 2×2 NoC architecture. Let us call this configuration our "basic NoC," which is considered here as the minimum search space for the detection of short faults. As will be discussed later in this paper, the test of larger NoCs can be defined based on the test of this basic NoC. We note that only interconnection faults are being tackled in this paper. Routers are assumed to be tested previously by using the strategy presented in [8] and cores and NIs are assumed to be tested afterward by using the method presented in [26].

For the basic NoC, there are 128 data wires of rr_links and rc_links, which can potentially be in short with each other (16 links $\times$ 8 data bits). From (1), there are 8,128 faults that will be tested in the circuit when considering pairwise shorts between only data wires. This number increases to 12,720 when considering the data and the control wires of bop and eop. Finally, when wires of the communication handshake are also considered, the number of faults grows up to 18,336.

3.2 A BIST-Based Approach

The problem of testing interconnects against short faults has been extensively studied and test sequences of minimum length for 100 percent detection and diagnosis have been reported [19], [20], [21]. The majority of the works that aim at detecting faults in the wires connecting two adjacent routers has proposed the insertion of BIST blocks in the routers [12], [22], as shown in Fig. 4a. However, by placing the BIST blocks into the routers, some faults in the rc_links and rr_links are not detected, as we will show next.

Let us assume two BIST blocks—TDG and test response analyzer (TRA)—at each of the four RaSoC routers composing the faulty neighborhood considered here, i.e., an 8-bit data 2×2 SoCIN network. TDG generates a test

sequence that will be injected in the channel, whereas TRA verifies whether the incoming data corresponds to the same sequence. When a crosstalk-specific sequence is used [12], we have measured a maximum fault coverage of 76.8 percent when all wires in each channel (data, control, and handshake ones) are subject to a fault (AND-short model), as mentioned. The test time for this sequence is 88.3 ms. By using a test sequence specific to short faults, such as the Walking-One sequence, still, the fault coverage is only 83 percent, but the test time is only 5.8 ms. Using random vectors with the same size of the Walking-One sequence, the coverage is also 76.8 percent (all wires eventually received values "1" and "0").

For this reason, we suggest that both the TDG and TRA are placed in such a way that faults in the wires connecting the core (through the NI) to the router and wires between distinct channels are also exercised during test, as shown in Fig. 4b. Hereafter, we assume that the NI is adapted to include the test structures (one TDG, one TRA, and multiplexers) that are used in the test mode. We note that, during test, only the NIs are set to the test mode, while the routers are kept in the normal mode.

4 The Proposed Interconnect Testing Approach

We propose a technique that is capable of detecting short faults between any two wires ($k = 2$ in (1)) of any channel at any direction of a regular 2×2 NoC implemented in a 2D mesh topology. The method is scalable with the NoC size and the links width, as will be shown in Section 7. We consider a 2D mesh NoC architecture for its known advantages in terms of regularity, concurrent data transmissions, and controlled electrical parameters [23]. Moreover, most industrial and academic NoC architectures presented so far are based on this topology [4], [15].

The proposed method consists of detecting all pairwise short faults of a 2×2 NoC topology by sending packets through this network. The challenges are 1) to define a test sequence that will be applied to this NoC topology so that 100 percent of the modeled faults are detected and 2) to define a packet flow that minimizes the test time. There are some important considerations that must be accounted for.

Consideration 1. Assuming the routers in the functional mode, the communication protocol (packets organized as header, payload, and tail) must be followed in order to apply test vectors to the NoC interconnects. The test sequence is placed on the packet payload and the header is mandatory to apply the payload. The contents of the header depend on the NoC and on the routing algorithm.

Consideration 2. Faults affecting the control wires (bop and eop) and the handshake signals (val and ack) must be tackled, independent of the faults affecting only data wires, since they have less controllability when the routers are in the functional mode.

Consideration 3. For short faults between any pair among the 16^*w ($w =$ the number of interconnect wires in the link) wires under test, one can think of the NoC as a large (16^*w)-bit wide bus. Consequently, the wires of all NoC channels must be completely filled with the test vectors in order to detect each fault. In other words, the packets must be sent in such a way that all channels are filled up at the same time.

Consideration 4. TDGs and TRAs are assumed to be connected to the NIs to send and receive the test packets similar to the basic BIST approach shown in Fig. 4b.

Consideration 5. Since all packets are sent with headers and tails, short faults may affect those flits, thus modifying the packet routing. When this occurs, the packet can be routed to any other node of the NoC or a signal of eop may not be received. In both cases, the TRA at the target node will not receive the packet or the eop within a predefined time interval, thus reporting a time-out fault. It is possible to perform the timing synchronization between all nodes because the transmission time of each packet is well known, as detailed in Sections 4.1 and 4.2. If the short fault does not cause packet loss, the packet payload is received by the target node and is checked by the TRA. In case of fault detection, the fault is classified as a payload fault. Since the network is in the normal mode of operation during test, one might consider the possibility of other errors caused by the short fault, such as a deadlock, in addition to packet losses and payload errors. We note that a fault cannot cause a deadlock within the basic NoC configuration because of the packets format and paths established during testing.

Consideration 6. The packet payload must contain all of the input vectors necessary for detecting the faults. Without loss of generality, we will exemplify our interconnect test approach by using the Walking-One Sequence [21] and evaluate its coverage to the AND-short and OR-short fault models. This sequence can be easily generated by hardware or software and presents maximum fault detectability of AND-shorts and OR-shorts. For other sequences and fault models, a similar analysis can be easily performed. The number of test vectors of the Walking-One Sequence is equal to the entire number of wires under test. As detailed in [24], although the headers alone can give a fault coverage of 34 percent, the complete Walking-One sequence must still be applied to achieve a 100 percent fault coverage.

Fig. 3 illustrates the application of the test sequence in the basic NoC topology. The proposed approach exercises all of the wires by using the router's functional mode. Considering the XY routing strategy, this means that four packets can be sent across the 2×2 mesh network. The paths followed by those packets are shown in Fig. 3 by the four different lines: One is solid, one is dashed, one has lines and points, and one has triple lines. The set composed of core i, $0 \leq i \leq 3$, and the associated NI (including TDG and TRA) is called node i from now on. Each line pattern represents one routing path:

- Path 00 (solid lines): one step east and one step south, from node 0 to routers 00, 01, and 11 to node 3.
- Path 01 (lines with points): one step west and one step south, from node 1 to routers 01, 00, and 10 to node 2.
- Path 10 (dashed lines): one step east and one step north, from node 2 to routers 10, 11, and 01 to node 1.
- Path 11 (triple lines): one step west and one step north, from node 3 to routers 11, 10, and 00 to node 0.

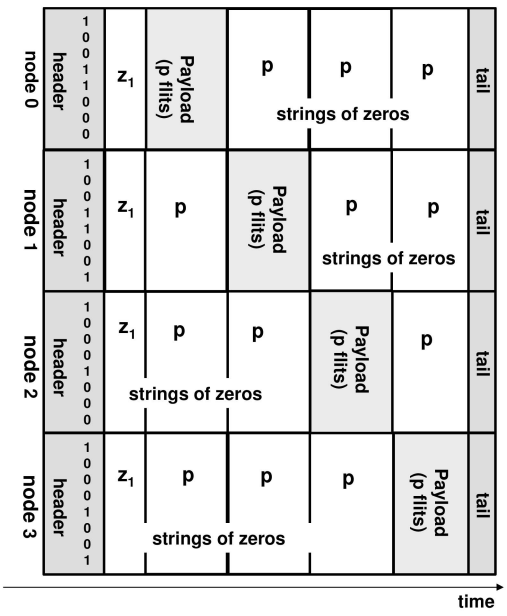

Fig. 5. Test sequence application for the detection of AND-shorts and OR-shorts in the Interconnect Data wires.

4.1 Tackling Faults in the Interconnect Data Wires

In the 2×2 NoC, the 128 wires under test are distributed into 16 8-bit links and only four paths can be implemented using the XY routing policy (see Fig. 3). Each path has four 8-bit data links that can be seen as a large 32-bit wide data bus. Thus, using the Walking-One Sequence for the basic NoC, we have 32 test vectors for each path. These 32 test vectors must then be organized into packets to be sent through the network.

Fig. 5 shows the organization of the four packets that are sent simultaneously by the four nodes. Note that, although all nodes use the same test sequence (32 vectors), the payload is shifted in time, so there is only one flit that contains only one bit at "1" at a time in the 128 data wires to ensure the detection capabilities of the Walking-One Sequence. Nodes are numbered from 0 to 3 and node 0 is assumed to be the first one to send test vectors, as shown in Fig. 5.

The detailed packet organization used in the proposed approach is shown in Fig. 6. There are 32 test vectors that must be sent as payload information flits, led by the header flit and followed by a tail in each packet. Each packet has five parts, assembled as follows:

Part 1. The header, defined according to the network routing strategy, contains routing information. Let us define the number of flits of the header as h.

Part 2. To ensure that only one bit is set to "1" at a time, after the header is sent, flits containing strings of zeros must be sent to reset all wires. Let us call this number z_1, which is equal to the number of cycles required to transmit the header from the source node to the target one.

Part 3. To shift the test vector application, additional zero strings must be included. Let us call z_2 the number of additional zero flits in this part of the packet. For node 0, no additional flits are necessary. For the remaining nodes, z_2 increases according to the node number i $(0 \leq i \leq 3)$ and the payload size (p), as shown in (2) and (3) (for w and z_3 defined in Part 4):

$$p = w.(1 + z_3), \tag{2}$$

$$z_2 = i.p. \tag{3}$$

Part 4. This part refers to the payload containing w test vectors for w the number of data wires in the link. A test vector is a single flit containing a single bit set to "1." Again, strings of zeros must be sent after each test vector in the payload to ensure that only one wire in the path is set to "1" at a time. The number of flits with zeros between vectors is equal to the number of clock cycles needed to send a payload flit from the source to the target node. Let us call this number z_3. The total number of flits in the payload (p) is thus $w.(1 + z_3)$, as shown in (2).

Part 5. Again, additional flits of zeros must be included to ensure that all wires are filled up during test while a test vector is being transmitted. According to node number i and the payload size p, the number of flits of zeros in this part of the packet is $(3 - i).p$.

Part 6. This part refers to the tail, defined according to the network protocol. Let us call the number of flits in the tail t (usually, $t = 1$).

Equation (4) defines the size of the test packet according to the description above. The resulting test time (in number of cycles) for the defined test sequence is given by (5) (for L, the latency for the test packet to arrive at the target node):

$$S = h + z_1 + 4.p + t, \tag{4}$$

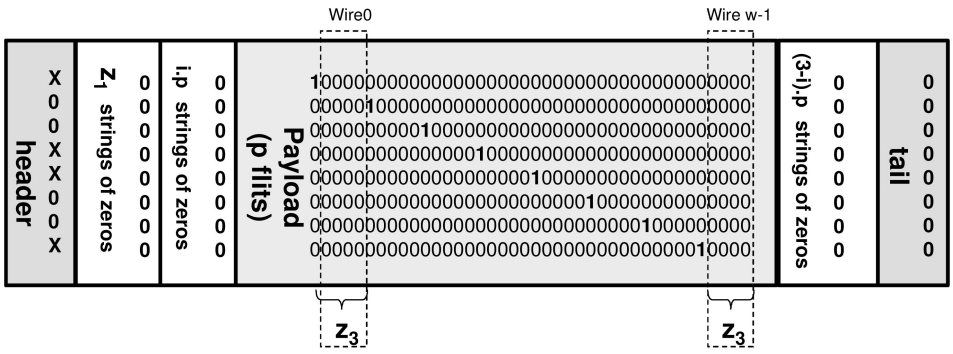

Fig. 6. Packet organization for the Walking-One Sequence for the detection of AND-shorts and OR-shorts in the Interconnect Data wires.

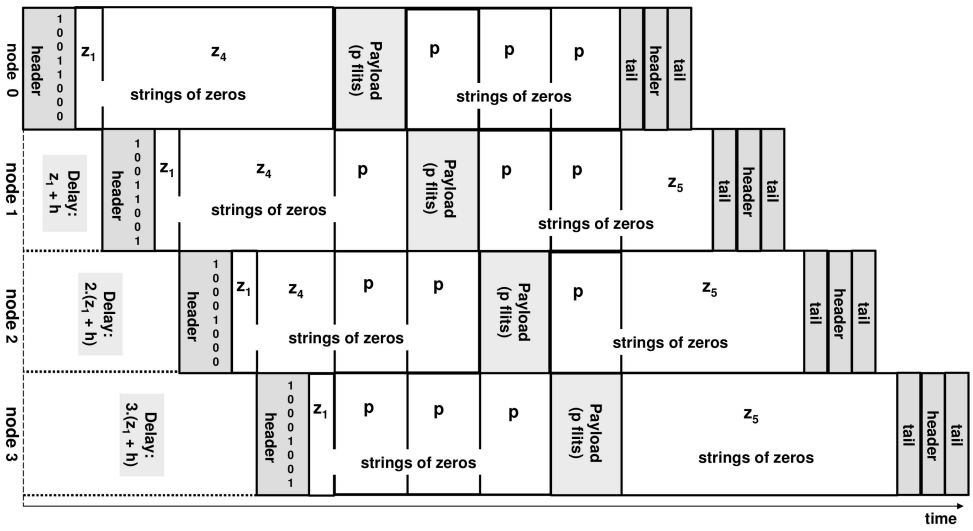

Fig. 7. Test sequence application for the detection of AND-shorts and OR-shorts in the Interconnect Data, Control, and Communication wires.

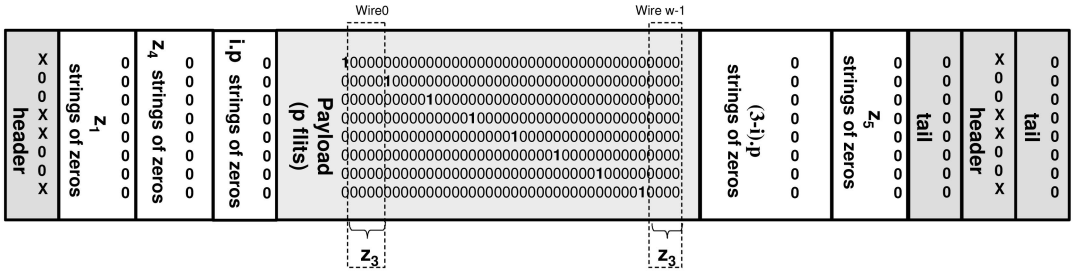

Fig. 8. Packet organization for the Walking-One Sequence for the detection of AND-shorts and OR-shorts in the Interconnect Data, Control, and Communication wires.

$$Total\ test\ time = S + L. \quad (5)$$

Thus, the packet size depends on the header and tail sizes, the number of cycles required to transmit the header, the number of vectors of the test sequence needed for a given number of wires in the link, and the number of cycles required to transmit a payload flit.

All nodes in the basic NoC send packets of the same size, but with the payload shifted in time. For the case study network, $h = 1$, $z_1 = 9$, $z_3 = 4$, $t = 1$, $w = 8$, and $L = 11$, which gives a total of 171 flits per test packet and a test time of 182 clock cycles for the entire 2×2 mesh NoC.

4.2 Tackling Faults in the Interconnect Control and Communication Wires

Let us include faults in the data, control framing bits (bop and eop wires), and communication handshake protocol bits (ack and val) in the proposed test strategy. Now, there are 192 wires that can potentially be upset. The same test sequence described before was applied in the network considering the new fault set.

The results have shown that some faults in the control wires (bop and eop) are not detected when all test packets are sent simultaneously. Indeed, since the four packets have all the same size and their transmissions of beginning and ending are all synchronized in the network, the bop and eop wires of both directions within a single channel hold the same values. Thus, faults between those wires are not detected. To solve this problem, the start time of each test sequence must be shifted in time. For instance, node 0 starts sending its sequence and, just after the header from node 0 is received by the target node, node 1 starts sending its header sequence, and so on, as illustrated in Fig. 7. We note that additional strings of zeros are added after the headers so that a header and a payload flit do not traverse the network at the same time.

In addition, some faults in the ack and val wires cannot be detected unless two packets are sent, one after another, through the same NoC path. The reason is that some of these faults can only be manifested when the routers are processing the tail of one packet and, at the next clock cycle, the header of a new packet. Thus, besides shifting the packet transmissions in time, a second packet must be added to the test sequence of each node. The second packet has only the header and tail flits since all other faults have been detected by the first packet. Extra strings of zeros must be added at the end of the first packet of each node to guarantee that a single tail or header is traversing the network at a time.

Fig. 7 shows the application of the new test sequence, while the new organization of the test packets is shown in Fig. 8. Each packet now has nine parts, assembled as follows:

Part 1. This part includes the header, as defined in Section 4.1.

Part 2. Before sending the payload, strings of zeros must be sent to fill up the channels until all headers achieve their

corresponding target nodes. Each header is separated by z_1 number of flits in zero.

Part 3. Additional zero strings are needed to shift the test vector application. So, z_4 is defined as the number of flits with zeros added after z_1 and before the payload. For each node, one needs as many flits as the number of clock cycles needed to send all other header flits from the source to the target node (z_1), as shown in the following:

$$z_4 = (3 - i).(z_1 + 1). \quad (6)$$

Part 4. This part includes the payload, as defined in (2).

Part 5. This part includes additional zero strings for filling up the channels after the payload, as defined in (3).

Part 6. Again, additional flits of zeros must be included to ensure that a single tail or header traverses the network at a time. Let us call the number of zero flits in this part z_5, which is defined by (7) according to the node number i and the number of clock cycles needed to send a header flit from the source to the target node z_1. The constant 3 in (7) comes from the tail plus header plus tail flits:

$$z_5 = i.(z_1 + 3). \quad (7)$$

Part 7. This part includes the tail, as defined in Section 4.1.

Part 8. This part includes the header, as defined in Section 4.1.

Part 9. This part includes another tail, as defined before.

Note that, now, the packets have different sizes and are sent at different times as well. Equation (8) defines the size of the test packet for the extended fault model according to the description above:

$$S_i = 2h + 4.z_1 + 2.i + 4.p + 2t + 3. \quad (8)$$

In the case studied, $h = 1$, $z_1 = 9$, $w = 8$, and $z_3 = 4$. So, for $i = 0$, the packet has 203 flits. For $i = 1$, the packet has 205 flits. For $i = 2$, the packet has 207 flits and, for $i = 3$, the packet has 209 flits.

The total test time for the basic NoC is given in (9), where the first term indicates the delay for the delivery of the header of node 3, the second term is the size of the test packet of node 3, and L is the latency for the packet transmission. In the case study circuit, the total test time is 250 clock cycles:

$$Total\ test\ time = 3(z_1 + h) + S_{i=3} + L. \quad (9)$$

5 Overall Test Structure

We assume that the test of the NoC interconnects is one part of the whole test of a NoC-based SoC, which consists of the test of the NoC and the test of the IP cores. The test of the NoC can be divided into the test of the routers and the test of the interconnections.

We assume, therefore, an overall test strategy as follows:

1. First, the NoC routers, the NIs, and the test structures (TDG and the TRA) are tested using a scan-based and external (vectors are delivered by the ATE) test strategy proposed in [8].
2. Then, the interconnections are tested by the method proposed in this work.
3. Finally, IP cores can be tested by reusing the NoC as a TAM, as proposed in [26].

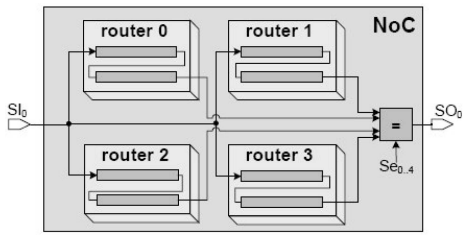

Fig. 9. Routers testing scheme [8].

The strategy proposed in [8] for the test of the routers considers the NoC as a flat core, i.e., a single test wrapper for the whole network is required. In addition, it does not require a full-scan implementation, which reduces the area overhead and uses the regular design of the NoC to reduce test time and data volume. A single scan chain using only the first position of the routers FIFOs is defined. This single scan chain provides the controllability and observability required to test the whole structure since the FIFO is usually not very deep and there is no feedback logic in this block. To complete the router testing, a second scan chain is defined with the remaining flip-flops of the control logic, which are the flip-flops used to implement the routing algorithm, for example. A generic test communication protocol, which can be applied to regular NoC topologies, is also proposed. In this protocol, test patterns coming from the external tester are simultaneously applied to all identical routers.

Test vectors are broadcast to routers by a single pin in the network interface and test responses of the routers are internally compared, as shown in Fig. 9 (only test-related ports and wires are shown). The block denoted by the equal signal indicates a comparator (one for each scan chain) that checks test responses against each other. Finally, an IEEE Standard 1500-compatible test wrapper ensures that the functional ports of the routers receive the same test stimuli, as detailed in [8].

The test of the embedded cores proposed in [26] assumes a fault-free network infrastructure that is reused as a TAM. First, the test vectors and test responses of each core are expressed as a set of packets to be transported through the network. In addition, the NI is modified to correctly send/receive the test data to/from the test interface of the core (scan controls, scan data pins, and functional pins). Finally, a comprehensive test scheduling algorithm defines a test sequence for all cores in the system while minimizing the overall test time.

5.1 The Test Structures

To implement the test strategy proposed in Section 4, TDGs and TRAs must be included in the logic that connects an IP core to the network, i.e., the network interface (NI). A programmable core can also perform those tasks. However, not all cores can be programmed and the network interface is designed according to the network protocol. Thus, the

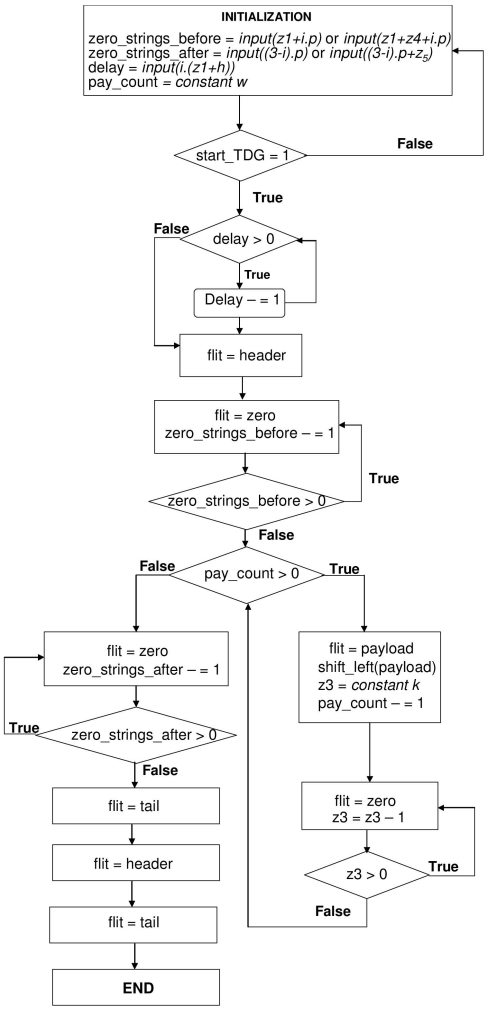

Fig. 10. Test data generator (TDG) logic for node i.

inclusion of the test structures in the NI is a more generic solution.

Figs. 10 and 11 present, respectively, the logic of the TDG and TRA that implement the test sequence proposed in Section 4.2.

Basically, the TDG generates the header, a number of zero flits followed by the payload flits, and, then, another series of zero flits followed by the tail. The numbers of flits of zeros depends on the node index and are inputs of the TDG. Those inputs are loaded from the ATE through a scan chain that connects all NIs. Data such as the link width w, the tail flit, and the number of cycles that a payload flit takes to traverse the network (z_3 as defined before) are constants.

Although, for the basic 2×2 NoC, all packets have a fixed target address, for larger NoCs, the actual address is variable, as will be explained in Section 7. Thus, the header data is also received from the ATE through the same scan chain. Another input of the TDG is a reference number (*delay* in Fig. 10) that defines the number of cycles that this node must wait before starting to send the test packet, as presented in Fig. 7. For the basic NoC, this delay value ranges from 0 to $3.(h + z_1)$.

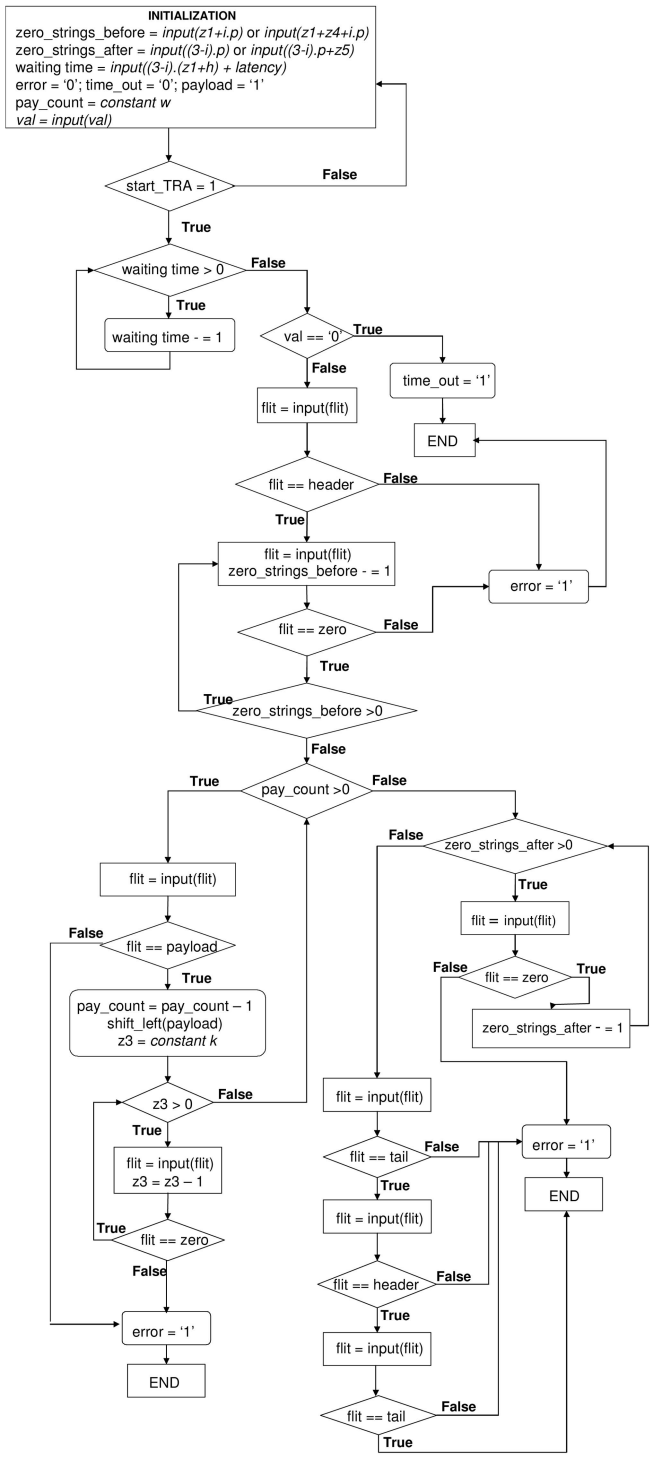

Fig. 11. Test response analyzer (TRA) logic for node i.

The TRA has a similar structure. The TRA waits for the val signal from the router to start the verification of the arriving data. If the val signal is not set within a predefined time interval (*waiting time* in Fig. 11), a time-out signal is given (the *time_out* flag is set), indicating a fault. If no time-out fault is detected, at each clock cycle, a new flit is read and TRA reproduces the expected data flits. Since control framing wires are also stored in the routers FIFOs, header and tail flits are also compared. Comparisons are based on data coming from the ATE (the same as for the TDG) and

on internal constants (modified header, tail, z_3, and w). The TRA generates two 1-bit flags—*time-out* and *error*—and makes them accessible to the ATE through the scan chain.

The ATE programs each TRA with a specific maximum *waiting time* for the packet reception. For instance, the TRA at node 0 is programmed with a *waiting time* based on the delay of the packet sent by node 3 and the network latency. On the other hand, the TRA at node 3 is programmed with a *waiting time* based on the delay of the packet sent by node 0 and the network latency. In summary, the TRA at node i is programmed with a waiting time of $(3-i).(z_1+h)+L$.

The test modules are included in each NI and a multiplexer is included in the NI output to connect the TDG outputs to the router during test. TRA inputs come from the router as well.

These two test structures were implemented in VHDL and synthesized using the Encounter RTL Compiler from Cadence by using a 0.35 technology library. The TDG and the TRA require, respectively, 378 and 454 equivalent gates to be implemented. Those two blocks represent 49 percent of the area of the RASoC router (1,688 equivalent gates). However, the routers alone have a rather small contribution to the whole system area, whereas the cores and the connections are responsible for most of the chip occupation. Therefore, the area overhead of the test structures does not have a great impact at the system level. Thus, we believe that the gains in terms of detection capabilities provided by the proposed implementation compensate for its cost.

5.2 TDG and TRA Configuration

The configuration of both the TDG and TRA is done through a scan chain connected to the ATE. Let us evaluate the length of the scan chain for each test module. According to the description in Section 4.2 and to the test sequence in Figs. 5 and 7, the maximum number of bits accounting for the zero flits previous to the payload and the zero flits after the payload will be the same for the TDG and the TRA and will be given by

$$zeros\_before\_payload(max) : \lceil(\log_2(z_1+3.w+3.w.z_3)\rceil, \tag{10}$$

$$zeros\_after\_payload(max) : \lceil\big(\log_2(3.w+3.w.z_3)\big\rceil. \tag{11}$$

For the TDG configuration, the following additional bits are needed in the scan register:

- Start_TDG: 1 bit.
- Header: $w+2$ bits.
- Delay (max): $\lceil\log_2(3.h+3.z_1)\rceil$.

For the TRA configuration, additional bits are also needed in the scan register:

- Start_TRA: 1 bit.
- Waiting time (max): $\lceil\log_2(3.z_1+3.h+L)\rceil$.
- Time-out flag: 1 bit.
- Error flag: 1 bit.

Equations (12) and (13) give the lengths (as a function of w) of the TDG and TRA scan chains, assuming that $z1=9$, $z3=4$, $h=1$, and $L=11$, as it applies to our NoC and our test sequences:

$$\mathrm{scTDG} = 8 + w + \lceil\log_2(9+15.w)\rceil + \lceil\log_2(15.w)\rceil, \tag{12}$$

$$\mathrm{scTRA} = 9 + \lceil\log_2(9+15.w)\rceil + \lceil\log_2(15.w)\rceil. \tag{13}$$

5.3 TDG and TRA Testing

The functional testing of the test modules results in a fault coverage of 80 percent for TDG and only 50 percent for TRA. Furthermore, an undetected fault in one of these modules, mainly TRA, can not only mask some interconnect faults (for instance, a stuck-at 0 in the error signal of TRA) but can also indicate an inexistent fault (a stuck-at 1 in the error signal of TRA). Thus, we assume in this work that TDGs and TRAs are tested a priori, together with the network interface, by using a scan-based method similar to the strategy defined for the routers. As the number of flip-flops in those logic is very reduced (83 flip-flops for TDG and 79 for TRA), they can also be tested using a full-scan approach to avoid any fault masking during the interconnections testing.

6 Experimental Results

In order to demonstrate the effectiveness of the proposed approach, a 2×2 SoCin NoC was implemented in VHDL, together with a fault injection mechanism that is capable of inserting AND-shorts and OR-shorts between all pair of wires in the design (data, framing, i.e., bop and eop, and handshake bits). Test sequence generators and TRAs were implemented as testbenches to speed up the simulation.

We note that, although our experiments were performed for the Walking-One Sequence, a similar analysis can be performed for any test sequence generator and its corresponding analyzer according to the defined fault model. The simulations were all performed using the ModelSim tool, where internal signals can be connected and disconnected using the command signal spy *force* and *release* in the testbench file. The entire simulation time takes only a few minutes.

Table 1 presents the results of the fault simulation campaign. Faults are classified as "detected by time-out only" (column 1), "detected by payload only" (column 2), and "detected by time-out and payload" (column 3).

First, the 8,128 faults were exhaustively injected in all 128 data interconnects of the 2×2 NoC topology. Lines 1 and 2 in Table 1 show the results for AND-shorts and OR-shorts, respectively, in data lines only, when the complete Walking-One Sequence is sent in each packet and the four packets are sent simultaneously. For each fault, only one detection is accounted for. The Walking-One Sequence is able to detect all faults in only 182 clock cycles by exercising all the functional NoC paths.

The second experiment considers the expanded fault set, where bop and eop wires are included, resulting in a total of 12,720 injected faults. Lines 3 and 4 in Table 1 show the results for AND-shorts and OR-shorts, respectively, when the complete Walking-One Sequence is sent in each packet and the four packets are sent simultaneously. Note that, as discussed previously in Section 4, very few faults (less than 1 percent) could not be detected. The undetected faults are the ones among the bop and eop lines at the same channel in opposite directions, where both wires hold the same

TABLE 1
Testing Fault Coverage Analysis

Experiment Setups Walking One Sequence	# clock cycles	Fault detection by			
		Time-out only	Payload error only	Payload error and Time-out	Total Detected
Faults in all channels data bits: 8,128 injected faults					
Packets sent at the same time AND-short faults	*182*	169 (2.08%)	6092 (74.95%)	1867 (22,97%)	8128 (100%)
Packets sent at the same time OR-short faults	*182*	307 (3.78%)	7616 (93.70%)	205 (2.52%)	8128 (100%)
Faults in all channels data and control bits: 12,720 injected faults					
Packets sent at the same time AND-short faults	*182*	2860 (22.48%)	5726 (45.01%)	4086 (32.12%)	12672 (99,62%)
Packets sent at the same time OR-short faults	*182*	1273 (10.00%)	9812 (77.02%)	1539 (12.09%)	12624 (99.24%)
Time shifted packets AND-short faults	*250*	3318 (24.51%)	5218 (41.02%)	4384 (34.46%)	12720 (100%)
Time shifted packets OR-short faults	*250*	1173 (9.22%)	10240 (80.50%)	1307 (10.28%)	12720 (100%)
Faults in all channels data, control and handshake bits: 18,336 injected faults					
Time shifted packets AND-short faults	*250*	5057 (27.58%)	7049 (38.44%)	6218 (33.91%)	18324 (99.93%)
Time shifted packets OR-short faults	*250*	5212 (28.43%)	10501 (57.27%)	2623 (14.30%)	18336 (100%)

value. When these cases happen, short faults have no effect in the packet transmissions, so they do not affect the transmission at this time. The number is slightly different from AND-short and OR-short types of faults because the signals bop and eop are expected to be at "1" just for one clock cycle when they are working properly.

Finally, lines 5 and 6 show the results for AND-shorts and OR-shorts, respectively, when the complete Walking-One Sequence is sent in each packet and the four packets are sent shifted in time. For the final sequence, a 100 percent fault coverage is achieved, at the price of a 37 percent increase in test time. Despite the increase, test time remains very reasonable, considering the number of faults that this approach is able to detect.

When considering the bits of the communication handshake protocol, 18,336 faults were injected. For OR-short faults, 100 percent of detection is achieved. For the AND-short type of faults, only 12 out of 18,336 faults are not detected (0.07 percent). The undetected faults are the ones among the ack_in, val_in, ack_out, and val_out at the same channel, where both wires hold the same value. This fault in the OR-short causes a time-out error because the router input buffer cannot indicate when it is full since it cannot set the ack to zero. In this case, packets are lost and, consequently, the fault is detected. But, in the case of the AND-short, the fault, in fact, does not result in an error. Since it is a wire-AND between ack and val wires, once the ack goes to zero, the val goes to zero as well and vice-versa and zero is the fault-free value for both wires. It is important to notice that this small number of undetected faults (less than 0.1 percent) does not affect the correct functionality of the network.

7 Method Scalability

The 2×2 basic NoC topology was used to define the minimum test configuration topology for the detection of realistic pairwise short faults affecting an on-chip network layout neighborhood. If short faults inside the refereed neighborhood will be detected in larger networks, a set of test configurations based on 2×2 sub-NoCs must be implemented. The test configurations that can run in parallel are grouped in the same test round, so the test application time is kept at a minimum. For instance, assume a 5×5 NoC topology, as depicted in Fig. 12.

The first test round in Fig. 12a is obtained by fulfilling the NoC with the basic test configuration (gray areas identify basic test configurations). Figs. 12b, 12c, and 12d show the other test rounds (containing four test configurations each) obtained by appropriately sliding right and down the first test round. These four test rounds altogether cannot cover all possible short faults in the network but cover all those realistic layout-level pairwise shorts that can affect any two wires in all channels inside any possible two-by-two routers neighborhood in the network.

These test rounds can be easily obtained for any $m \times m$ NoC, $m \geq 3$, through the following procedure:

1. To obtain test round 1, fill in the whole NoC, from left to right and from top to bottom, using as many basic test configurations as possible, as shown in Fig. 12a.
2. Shift the test round obtained in Step 1 one column to the right to obtain test round 2 (Fig. 12b).
3. Shift the test round obtained in Step 1 one row to the bottom to obtain test round 2 (Fig. 12c).
4. To obtain test round 4, simultaneously shift test round 1 one column to the right and one row to the bottom, as shown in Fig. 12d.

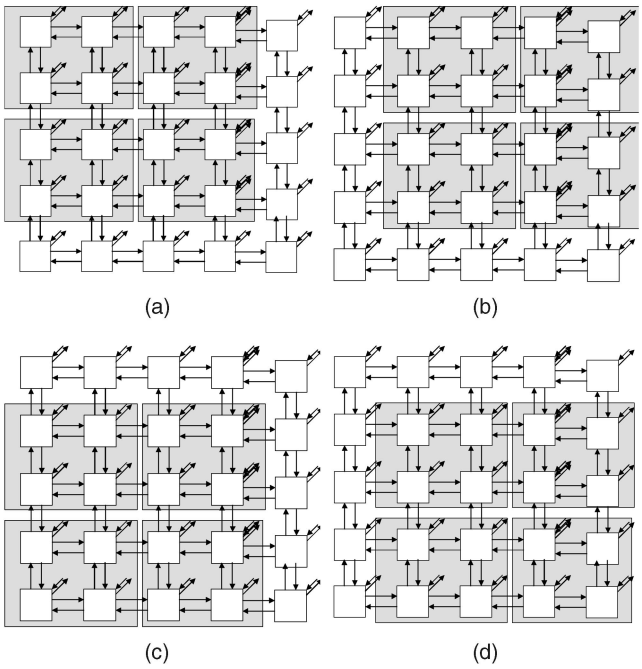

Fig. 12. The test rounds and respective test configurations for a 5×5 mesh NoC.

For the example in Fig. 12, 16 test configurations are applied in four test rounds. Notice that this procedure always leads to four test rounds for larger than 2×2 NoCs. As mentioned, the window of short fault possibilities is a 2×2 NoC; thus, the set of wires that are susceptible to pairwise shorts are the ones placed together at the layout level in such a neighborhood.

We assume that the probability of shorts beyond this limit is very low, as explained before. However, the designer could think of enlarging this short detection neighborhood by adding new test configurations to those proposed in this work. For example, our basic 2×2 NoC topology does not exercise $N \leftrightarrow S$ and $W \leftrightarrow E$ routing possibilities. In this case, a 3×3 neighborhood could be defined in such a way that these additional routing possibilities are checked by just adapting the interconnect test sequence proposed here. On the other hand, the designer can make the opposite decision and use fewer test configurations to test only some parts of his layout, i.e., those that are shown to be more susceptible to short faults in a particular design.

The number of wires of each interconnect link w may vary, even in the minimum test configuration topology. The curve in Fig. 13a shows the effect of w in the number of short faults, as given by (1), for $n = 16^{*}w$ and $k = 2$, that must be detected in the 2×2 NoC topology. As expected, the number of faults grows exponentially with the number

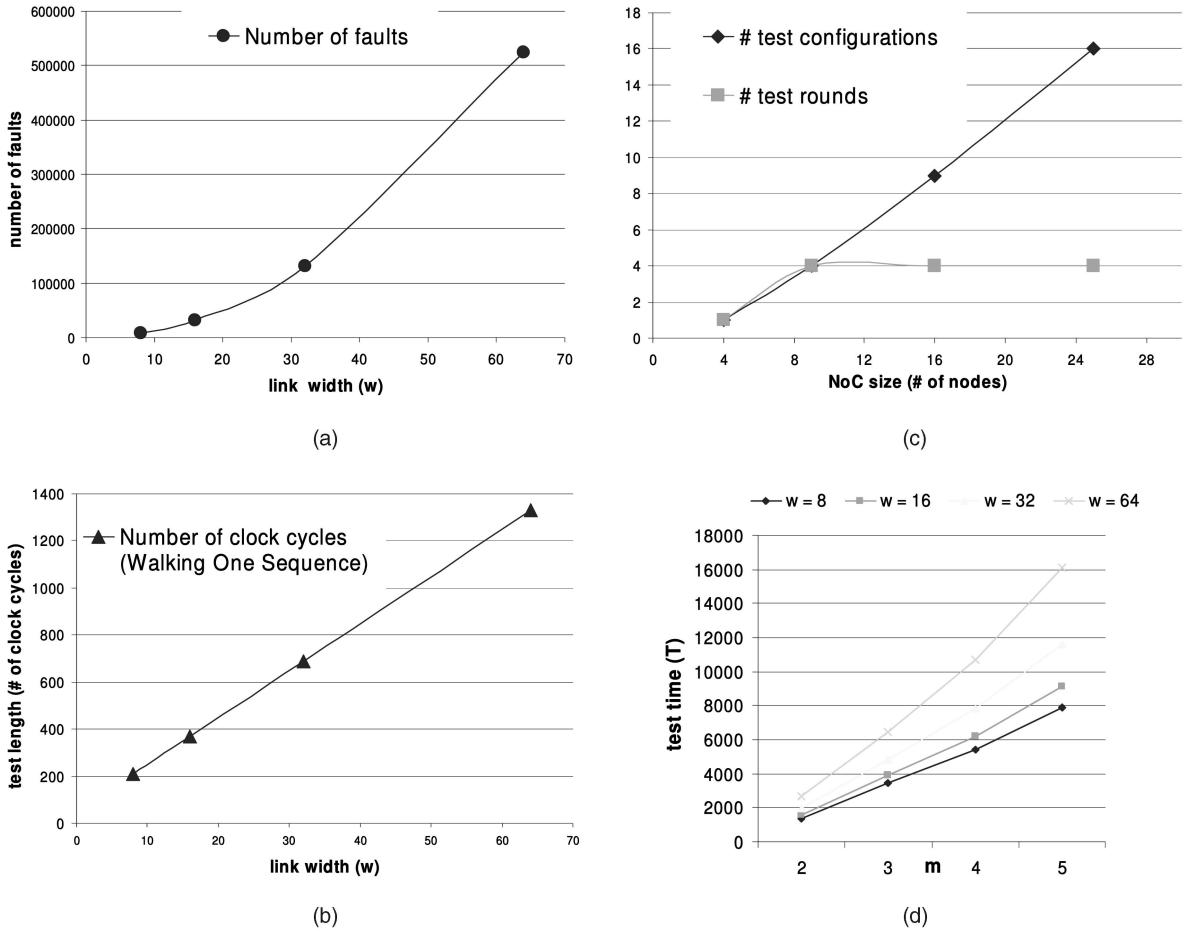

Fig. 13. Scalability of the proposed approach. (a) Link width w versus number of short faults. (b) Link width w versus number of test cycles. (c) Influence of the NoC size in the number of test configurations and rounds. (d) Test time for different values of w and m.

of wires in each link. On the other hand, the number of test clock cycles C needed for testing and given by (9) grows linearly in the proposed technique, as shown in the curves in Fig. 13b. Considering the size of the network, Fig. 13c shows the number of test configurations and the number of test rounds to test 2×2, 3×3, 4×4, and 5×5 NoC topologies.

For an $m \times m$ NoC, $m \geq 3$, the number of test configurations (tc) is given by

$$\mathrm{tc} = m^2 - 2.m + 1. \quad (14)$$

The number of short faults that the proposed approach can detect is calculated by multiplying the number of possible pairwise shorts in the 2×2 neighborhood (given by (1), for $n = 16^{*}w$ and $k = 2$) by the number of test configurations given by (14).

Although tc grows quadratically with m, according to the procedure discussed previously in this section, the number of test rounds tr for any $m \times m$ NoC, $m \geq 3$, is kept constant:

$$\mathrm{tr} = 4. \quad (15)$$

As discussed previously, this is due to the fact that many test configurations can be accommodated in the same test round and thus can be run in parallel.

Considering now that the configuration scan chains of all TDGs and TRAs of all NIs are placed together in a single communication chain sc with the ATE, the length of the register used for the reconfiguration of one node (one pair of TDG and TRA) is given by

$$\mathrm{sc} = 17 + w + 2.\lceil \log_2(9 + 15.w) \rceil + 2.\lceil \log_2(15.w) \rceil. \quad (16)$$

The total NoC interconnect test application time will be given by

$$T = 2.m^2.\mathrm{sc} + C, \quad \text{for } m = 2, \quad (17)$$

$$T = (\mathrm{tr} + 1).(m^2.sc) + \mathrm{tr}.C, \quad \text{for } m > 2, \quad (18)$$

where $tr = 4$, m^2 is the number of cores in an $m \times m$ NoC, sc is the scan length of the register used for the reconfiguration of the TDG and TRA, and C (the number of test cycles required by the sequence) is given by (5) or (9), depending on the desired fault coverage. It can be noticed that the test application time due to the TDG and TRA configuration grows quadratically with the size of the NoC and linearly with the scan chain length, while the time due to the test sequence itself keeps constant.

Calculating sc for $w = 8$, 16, 32, and 64 gives 55, 65, 85, and 121, respectively. With these scan lengths and for $C = 250$ clock cycles, as given by (9), it is clear from (17) and (18) that the test application time contribution due to the TDG and TRA configuration dominates T. Fig. 13d plots (17) and (18) for different values of w (as above) and m, testifying to this behavior. On the one hand, one can conclude from these results that the test sequence proposed and the way that it scales for larger NoCs (number of test rounds) are very efficient. On the other hand, it is clear that there is still room for reducing the total test application time by optimizing the way that TDGs and TRAs are configured by the external ATE.

8 Conclusions and Future Works

We have proposed, implemented, and analyzed a new strategy for at-speed testing of interconnect channels in mesh NoCs considering an extended fault model for short faults. Current work includes the analysis of alternative implementations of TDGs and TRAs to evaluate its area overhead and testability. In addition, we are now working on an integrated method for testing both the routers and the interconnections by using some ideas presented in this paper.

Acknowledgments

This work was supported in part by the Conselho Nacional de Desenvolvimento Cientifico e Tecnológico (CNPq) and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES).

References

[1] F.G. Moraes, N. Calazans, A. Mello, L. Möller, and L. Ost, "HERMES: An Infrastructure for Low Area Overhead Packet-Switching Networks on Chip," *Integration: The VLSI J.*, vol. 28, no. 1, pp. 69-93, 2004.

[2] P. Guerrier and A. Greiner, "A Generic Architecture for On-Chip Packet-Switched Interconnections," *Proc. Conf. Design, Automation and Test in Europe,* pp. 250-256, 2000.

[3] C. Zeferino and A. Susin, "SoCIN: A Parametric and Scalable Network-on-Chip," *Proc. 16th Symp. Integrated Circuits and Systems Design,* pp. 169-174, 2003.

[4] B. Vermeulen, J. Dielissen, K. Goossens, and C. Ciordas, "Bringing Communication Networks on a Chip: Test and Verification Implications," *IEEE Comm. Magazine,* vol. 41, no. 9, pp. 74-81, Sept. 2003.

[5] E. Cota, L. Carro, and M. Lubaszewski, "Reusing an On-Chip Network for the Test of Core-Based Systems," *ACM Trans. Design Automation of Electronic Systems,* vol. 9, no. 4, pp. 471-499, 2004.

[6] J. Ahn and S. Kang, "Test Scheduling of NoC-Based SoCs Using Multiple Test Clocks," *ETRI J.*, vol. 28, no. 4, pp. 475-485, Aug. 2006.

[7] J. Kim et al., "On-Chip Network Based Embedded Core Testing," *Proc. IEEE Int'l SOC Conf.,* pp. 223-226, 2004.

[8] A.M. Amory, E. Briao, E. Cota, M. Lubaszewski, and F.G. Moraes, "A Scalable Test Strategy for Network-on-Chip Routers," *Proc. IEEE Int'l Test Conf.,* p. 9, 2005.

[9] K. Stewart and S. Tragoudas, "Interconnect Testing for Networks on Chips," *Proc. 24th IEEE VLSI Test Symp.,* p. 6, 2006.

[10] C. Grecu, P. Pande, B. Wang, A. Ivanov, and R. Saleh, "Methodologies and Algorithms for Testing Switch-Based NoC Interconnects," *Proc. 20th IEEE Int'l Symp. Defect and Fault Tolerance in VLSI Systems,* pp. 238-246, 2005.

[11] J. Raik, V. Govind, and R. Ubar, "An External Test Approach for Network-on-a-Chip Switches," *Proc. 15th Asian Test Symp.,* pp. 437-442, 2006.

[12] C. Grecu, P. Pande, A. Ivanov, and R. Saleh, "BIST for Network-on-Chip Interconnect Infrastructures," *Proc. 24th IEEE VLSI Test Symp.,* p. 6, 2006.

[13] P.P. Pande, A. Ganguly, B. Feero, B. Belzer, and C. Grecu, "Design of Low Power and Reliable Networks on Chip through Joint Crosstalk Avoidance and Forward Error Correction Coding," *Proc. 21st IEEE Int'l Symp. Defect and Fault Tolerance in VLSI Systems,* pp. 466-476, 2006.

[14] F. Angiolini, P. Meloni, S. Carta, L. Benini, and L. Raffo, "Contrasting a NoC and a Traditional Interconnect Fabric with Layout Awareness," *Proc. Int'l Conf. Design, Automation and Test in Europe,* pp. 1-6, 2006.

[15] J. Kim, C. Nicopoulos, D. Park, M. Yousif, N. Vijaykrishnan, and C. Das, "A Gracefully Degrading and Energy-Efficient Modular Router Architecture for On-Chip Networks," *Proc. 33rd Ann. Int'l Symp. Computer Architecture,* pp. 4-15, 2006.

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

[17] L. Benini and G. De Micheli, "Networks on Chips: A New SoC Paradigm," *Computer*, vol. 35, no. 1, Jan. 2002.
[18] C.A. Zeferino, M.E. Kreutz, and A.A. Susin, "RASoC: A Router Soft-Core for Networks-on-Chip," *Proc. Int'l Conf. Design, Automation and Test in Europe Conf.*, pp. 198-203, 2004.
[19] W. Kautz, "Testing for Faults in Wiring Networks," *IEEE Trans. Computers*, vol. 23, no. 4, pp. 358-363, Apr. 1974.
[20] W.-T. Cheng, J.L. Lewandowski, and E. Wu, "Diagnosis for Wiring Networks," *Proc. IEEE Int'l Test Conf.*, pp. 565-571, 1990.
[21] C. Stroud, "A Designer's Guide to Built-In Self-Test," *Frontiers in Electronic Testing*, vol. 19, Springer, 2002.
[22] T. Bengtsson, A. Jutman, S. Kumar, R. Ubar, and Z. Peng, "Off-Line Testing of Delay Faults in NoC Interconnects," *Proc. Ninth EUROMICRO Conf. Digital System Design: Architectures, Methods and Tools*, pp. 677-680, 2006.
[23] R. Holsmark and S. Kumar, "Design Issues and Performance Evaluation of Mesh NoC with Regions," *Proc. NORCHIP Conf.*, pp. 40-43, 2005.
[24] E. Cota, F.L. Kastensmidt, M. Cassel, P. Meirelles, A. Amory, and M. Lubaszewski, "Redefining and Testing Interconnect Faults in Mesh NoCs," *Proc. IEEE Int'l Test Conf.*, 2007.
[25] S. Murali, P. Meloni, F. Angiolinit, D. Atienzat, S. Cartas, L. Benini, G. De Michelit, and L. Raffo, "Designing Application-Specific Networks on Chips with Floorplan Information," *Proc. IEEE/ACM Int'l Conf. Computer-Aided Design*, pp. 355-362, 2006.
[26] E. Cota and C. Liu, "Constraint-Driven Test Scheduling for NoC-Based Systems," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 11, pp. 2465-2478, Nov. 2006.

Érika Cota received the BS degree in computer science from the Federal University of Minas Gerais and the MS and PhD degrees in computer science from the Federal University of Rio Grande do Sul. She is currently an adjunct professor in the Department of Computer Science, Federal University of Rio Grande do Sul. Her research interests include testing, DFT, and fault tolerance of hardware and software embedded systems. She is a member of the IEEE.

Fernanda Lima Kastensmidt received the BS degree in electrical engineering and the MS and PhD degrees in computer science from the Federal University of Rio Grande do Sul. She is currently a professor of computer science at the Federal University of Rio Grande do Sul. Her research interests include VLSI testing and design, fault effects, fault-tolerant techniques, and programmable architectures. She is a member of the IEEE.

Maico Cassel is currently an undergraduate student of electrical engineering at the Federal University of Rio Grande do Sul. He was an undergraduate researcher for two years and is currently an intern at NSCAD Microelectronics. His research interests include adaptative systems, NoCs, and DSP.

Marcos Hervé received the BS degree in electrical engineering from the Federal University of Rio Grande do Sul, where he is currently an MSc student. His research interests include VLSI design and testing, DFT, and networks-on-chip.

Pedro Almeida is currently an undergraduate student in computer science at the Federal University of Rio Grande do Sul. His research interests include fault-tolerant techniques and digital circuit design.

Paulo Meirelles received the bachelor's degree in computer software development technology from the Federal Technological Education Center of Rio Grande do Norte (CEFET-RN). He is currently working toward the master's degree in computer science at the Federal University of Rio Grande do Sul. His research interests include software engineering for embedded systems and embedded software testing.

Alexandre Amory received the PhD degree from the Federal University of Rio Grande do Sul in 2007. His PhD research was focused on the testability of SoCs based on networks-on-chip. He is currently an ASIC verification engineer at the CEITEC Design House.

Marcelo Lubaszewski received the bachelor's degree in electrical engineering and the MSc degree from the Federal University of Rio Grande do Sul (UFRGS) in 1986 and 1990, respectively, and the PhD degree from the Institut National Polytechnique de Grenoble in 1994. He is currently with UFRGS, where he has been an associate professor since 1990. His research interests include the design and test of mixed-signal, microelectromechanical, core-based, and NoC-based systems, self-checking and fault-tolerant architectures, and computer-aided testing. He is a member of the IEEE and the IEEE Computer Society.