Multi-Layer Test and Diagnosis for Dependable NoCs

Wunderlich, Hans-Joachim; Radetzki, Martin

Proceedings of the 9th IEEE/ACM International Symposium on Networks-on-Chip (NOCS'15) Vancouver, BC, Canada, 28-30 September 2015

doi: http://dx.doi.org/10.1145/2786572.2788708

Abstract: Networks-on-chip are inherently fault tolerant or at least gracefully degradable as both, connectivity and amount of resources, provide some useful redundancy. These properties can only be exploited extensively if test and diagnosis techniques support fault detection and error containment in an optimized way. On the one hand, all faulty components have to be isolated, and on the other hand, remaining fault-free functionalities have to be kept operational. In this contribution, behavioral end-to-end error detection is considered together with functional test methods for switches and gate level diagnosis to locate and to isolate faults in the network in an efficient way with low time overhead.

Multi-Layer Test and Diagnosis for Dependable NoCs

Hans-Joachim Wunderlich
Computer Architecture
University of Stuttgart
Pfaffenwaldring 47, D-70569 Stuttgart
wu@informatik.uni-stuttgart.de

Martin Radetzki
Embedded Systems Engineering
University of Stuttgart
Pfaffenwaldring 5b, D-70569 Stuttgart
radetzki@informatik.uni-stuttgart.de

ABSTRACT

Networks-on-chip are inherently fault tolerant or at least gracefully degradable as both, connectivity and amount of resources, provide some useful redundancy. These properties can only be exploited extensively if test and diagnosis techniques support fault detection and error containment in an optimized way. On the one hand, all faulty components have to be isolated, and on the other hand, remaining fault-free functionalities have to be kept operational.

In this contribution, behavioral end-to-end error detection is considered together with functional test methods for switches and gate level diagnosis to locate and to isolate faults in the network in an efficient way with low time overhead.

Categories and Subject Descriptors

B.4.5 [**Input/Output and Data Communications**]: Reliability, Testing, and Fault-Tolerance – *built-in tests, diagnostics.*

General Terms

Performance, Design, Reliability

Keywords

Test, diagnosis, fault tolerance, network-on-chip, cross-layer.

1. INTRODUCTION & RELATED WORK

The inherent fault tolerance of networks-on-chips (NoCs) makes them a special candidate to cope with the reliability threats that accompany further CMOS scaling [25]. While the "power wall" limits the frequency increase and enforces performance improvements by exploiting parallelism, the resulting "reliability wall" can only be overcome efficiently by applying test and diagnosis schemes at the various network layers of an NoC.

High quality test and diagnosis schemes are technology dependent, and a purely functional approach is not sufficient for reaching the same quality as obtained by structural techniques. The abstraction levels of fault model are related to some extent to the network layer definition of the ISO/IEC 7498-1:1994 OSI seven layer model.

1.1 Physical Layer

Defects consist of additional, missing or wrong physical material, and they are modeled by *faults* of a structural gate level circuit model. Standard fault models include stuck-at faults, transition faults, delay faults, crosstalk or various types of bridging faults. They are associated in this paper with the physical network layer, and require the classical structural methods of automated test pattern generation (ATPG) [5] and test application through test access mechanisms (TAM) such as scan chains [32].

NoC-specific adaptations of these methods include the optimization of scan structures according to NoC topology [14], the transport of test patterns to scan chains using flits [22], and standards-compliant test wrappers for NoC [3].

Beyond just identifying faulty circuits, the circuit's test response can be analyzed by structural diagnosis techniques to locate the faulty circuit component (net or logic gate). Diagnosis can be performed offline with automated test equipment or in situ with dedicated built-in self-test (BIST) logic.

The diagnosis result can be used offline (to increase production yield) or online (to cope with emerging faults) by repairing or deactivating faulty circuitry. Repair requires redundant circuit elements such as spare wires [21] to be designed in up-front whereas deactivation keeps the circuit alive at the cost of reduced functionality or performance (graceful degradation, e.g. through reduced flit size [31]).

1.2 Data Link Layer

On the data link layer, which establishes connectivity and flow control between adjacent switches, these classical structural test methods are not anymore directly applicable as both, pattern generation and pattern application, are constrained to a well-formed format for data transmission between two switches. On one hand these constraints reduce the reachable fault coverage, on the other hand overtesting is avoided and tests can be executed more efficiently.

An NoC-specific BIST architecture featuring a dedicated test controller and the usage of the NoC data links as TAM has been described by Grecu et al. [13]. Lehtonen et al. show how links can be reconfigured in order to cope with faults. A method for mapping diagnosed faults to switch ports [9] enables graceful degradation by deactivating defective ports and the connected links and is also used in the paper at hand.

An alternative to these diagnostic approaches is concurrent error detection and error correction. These techniques rely on the use of error correcting or detecting codes (ECC/EDC). Since respective codecs are required in each switch, cheap single error correcting (SEC) codes such as Hamming codes are employed. In case of EDC, switch-to-switch retransmission can be applied for correcting transient errors, but is not effective against permanent

NOCS '15, September 28 - 30, 2015, Vancouver, BC, Canada.
© 2015 ACM. ISBN 978-1-4503-3396-2/15/09…$15.00
DOI: http://dx.doi.org/10.1145/2786572.2788708

defects. Studies ([4][15]) show that the incurred area and power overhead is not justified unless extremely high failure rates are assumed, and suggest the application of such techniques on higher layers instead.

1.3 Network Layer

The network layer establishes functionalities of packet routing and switching in NoC switches (which include routing units). For testing and diagnosis, the circuit-level structural fault model is widely abstracted. For example, Kohler et al. [18] suggest a functional fault model (xbar faults) that captures connection paths in crossbar switches. Further abstracting, *functional failure modes* like misrouting or data corruption are used to capture the effect of low-level defects on switch functionalities ([2]).

When circuit-level structural diagnosis is applied to NoC switches, a mapping of diagnosed structural faults to the affected functionalities can be established [7]. Alternatively, functional tests can be applied [1]. In reverse, structural faults can be diagnosed with functional techniques [16], and SAT-based ATPG can be employed to ensure high structural coverage of functional software-based self-test (SBST) [10]. Like on other layers, concurrent error detection with error detecting codes (e.g. [18]) can replace or supplement diagnostic techniques. The use of fault-secure synthesis techniques [11] ensures that all faults manifest as detectable errors.

In order to achieve better graceful degradation than with a complete switch shutdown, defective parts of a switch can be bypassed by data path reconfiguration [23] or can be omitted by local routing adaptation [18]. Potentially resulting problems related to congestion or deadlocks can be avoided by ahead-looking adaptation of adjacent switches [27].

1.4 Transport Layer

Finally, the transport layer includes the end-to-end data transmissions from the original sender to the designated receiver. The use of error-detecting codes (EDC) such as parity (single error detecting), extended Hamming (double error detecting), or cyclic redundancy check (CRC, capable of detecting error bursts) is common for concurrent error detection on this layer. Alternatively, the use of heartbeat messages has been suggested [12], which replaces the overhead of equipping each packet with an EDC field by the potentially smaller overhead of eventual test packets. Also the use of forward error correcting codes (FEC) has been investigated [20], but the cost of decoding advanced codes with error-correcting capacity that goes beyond the single error correction (SEC) of Hamming codes, e.g. Reed-Solomon or BCH, appears prohibitive.

To diagnose NoCs on the network layer, Raik et al. [26] suggest a method that uses end-to-end messages injected and ejected at test access points at the boundaries of a mesh network. Zhang et al. [34] describe a software-based localization method that gathers information about the position of nodes that have been deactivated after an unsuccessful BIST run. Contrary to that, in Section 2 we outline a diagnostic method that locates defective NoC resources (links, switches) on the network layer using regular data packets.

1.5 Cross-Layer Methods

It is advantageous to separate monitoring and coarse fault diagnosis from the more expensive fine grained fault diagnosis for defect location, at least if we are dealing with low and medium error rates. Detecting faulty switches and links is targeted efficiently at the transport layer, while diagnosis for defect location needs finally structural information obtained by lowering the abstraction level in a top down fashion. This leads to a top-down divide-and-conquer approach across the network layers and will finally point to a defective structure, e.g. wire, port or gate.

However, the description of this proceeding is preferably done in a bottom-up way, layer for layer as functionalities and concepts can be reused this way. Hence, this paper is organized as follows: After describing test and monitoring at the transport layer in the next section, test, diagnosis and fault isolation at gate level are discussed in section 3. Section 4 introduces software based self-test at the data link layer, and section 5 presents the concept of functional failure modes at the network layer.

2. TRANSPORT LAYER

2.1 Transport Protocol

If the absence of post-manufacturing defects is a reasonable assumption, as still the case with current technology, a minimal transport protocol for packetization and re-assembly of end-to-end messages is sufficient. For future technologies, adaptive repeat request (ARQ) techniques can be employed for retransmission of erroneous packets. This requires each packet to be equipped with an error-detecting code (EDC).

To implement ARQ, a sender keeps a local copy of each sent packet until it is positively acknowledged by the receiver. Should the receiver detect an error by decoding the EDC, it sends a negative acknowledgement. Multiple acknowledgements can be bundled in a single protocol packet so as to reduce the incurred traffic overhead. Upon receiving a negative acknowledgement, the packet is re-sent. If the receiver is not capable of reordering packets, subsequent packets must also be retransmitted.

Since data packets may be completely lost, the receiver implements a time-out mechanism upon which expected packets that did not arrive are negatively acknowledged. Missing packets can be detected by gaps in the sequence IDs transmitted as part of the packet header. Acknowledgement messages may be lost as well. Therefore, the sender implements another time-out after which a yet unacknowledged packet is automatically re-sent.

2.2 Diagnostic Protocol

Retransmission is able to correct transient faults by temporal redundancy. However, in case of a permanent fault, deterministic routing would lead any retransmitted packet though the same defective component, where it is again corrupted. This situation can be detected with an error counter for failed retransmission attempts. A fault can thereby be classified as permanent, which leads to the need of locating it so as to change routing paths.

For this purpose, a scoreboard-based mechanism has been suggested [30] that narrows down fault location by using statistics of faults occurred on multiple transmission paths: Those network resources present in a maximal number of faulty paths are likely fault candidates. To overcome the probabilistic nature of this approach, we have proposed a bisection mechanism [28] to iteratively narrow down fault location to a single switch, using a single transmission path.

Our method assumes that the transport layer has some information on the routing policy that is implemented on the network layer: Namely, the path length be known and the switch in the middle of the path be identifiable. This is easily implemented for a deterministic routing scheme such as dimension order routing. Also table-based routing information, where routing table entries are computed in software by the processing elements, could be exploited.

Given sender node n_s and receiver node n_r, the middle node n_i is identified as intermediate node. The sender directs the packet at n_i and supplies the final target address of n_r in an additional header data field. At n_i, the packet is consumed and checked. If it is erroneous, it is negatively acknowledged and n_s further bisects the path by addressing the switch halfway to the previous intermediate node. Otherwise, the fault must be on the second half of the end-to-end path. Node n_i takes over the role as the new sender of the packet and continues with the same bisection protocol. The process repeats until eventually the fault location is narrowed down to a single network resource, namely the link between two adjacent switches. Should the fault reside inside one of the switches and affect multiple links, these links will also be identified faulty as soon as they make other transmissions fail.

2.3 Fault-Tolerant Routing

When a faulty resource is identified, the routing should be adapted to prevent further (re-)transmission using that resource. This can be achieved by triggering routing adaptation on the network layer [29]. For this purpose, information about the identified fault location would have to be passed down the layer hierarchy. In [28] we use a software re-routing approach that is implemented on the transport layer only.

For software routing, the original sender identifies an intermediate node so that the path to the intermediate node is not affected by faults. Similar to the diagnostic protocol, this requires intelligence on the routing policy. Moreover, each network interface (or its attached processing element) has to keep track of faults diagnosed on its packet transmissions. Packets are sent to the intermediate node and the final destination is encoded in the additional address field already implemented for diagnostic purposes. The intermediate node consumes the packet, replaces the intermediate address with the final address, and re-injects the packet. Should the intermediate node know of a fault on the regular path to the final destination, it chooses another intermediate instead.

The choice of intermediate nodes can be improved with global knowledge of the network state. This enables the sender to identify an intermediate node so that not only the path to the intermediate but also the path from intermediate to destination is free of known faults [17].

2.4 Assessment

The method described above can be classified as *online* and *concurrent*, that is, it is performed while the NoC is in operation and does not preempt the regular data traffic. The system thus remains operative, albeit with a certain performance loss due to the overhead for the diagnostic protocol, packet consumption and re-injection. This graceful degradation is a preferable alternative compared to system failure that would result from persistent packet loss. Moreover, the hardware overhead is limited: Whereas the network interface may need some additional hardware for timers and error counters, the protocol is implemented in software on the processing element. We also assume that memory space for retransmission and reordering buffers is allocated in the processing element's local memory that is shared with the network interface.

On the downside, diagnostic quality is rather limited:

- The granularity of fault location is coarse, on the level of network links and switches. More fine-grained diagnosis would offer the potential of reducing performance degradation.
- The method only diagnoses faults that manifest as observable errors, but not latent faults. For example, packets may be misrouted due to a defect in a switch, but still arrive intact at the destination. Eventually such latent fault can lead to a deadlock because the misrouting violates the otherwise deadlock-free routing policy.
- False positives can occur, i.e. the method may erroneously diagnose intact resources as faulty. This happens when congestion appears due to retransmission, acknowledgement and software routing overhead. In this case, positive acknowledgements may be delayed so much that the timeout mechanism lets the diagnostic protocol assume negative acknowledgement on intact path sections.

High diagnostic quality requires diagnosis on lower abstraction levels, closer to the physical failure mechanisms. Yet transport layer techniques are useful in narrowing down potential fault locations so that the more costly lower-layer diagnostic techniques can be constrained to a small section of the NoC. To this end, interaction and sharing of information is required among NoC layers:

- To achieve finer granularity, the transport layer should trigger detailed diagnosis on lower layer specifically for fault candidates.
- The lower layer diagnosis technique should give feedback on false positives so that resources that were erroneously diagnosed and deactivated on transport layer can be revived.
- The lower layer can also give feedback on identified latent faults so that the transport layer (or potentially, by routing adaptation, the network layer) is able to avoid the use of resources with latent problems.

3. PHYSICAL LAYER

The faults at the physical layer are described by a structural fault model at gate level. The faults at the interconnect lines between switches or between switches and cores include open lines, bridges, and delay, transition or crosstalk faults (Figure 1).

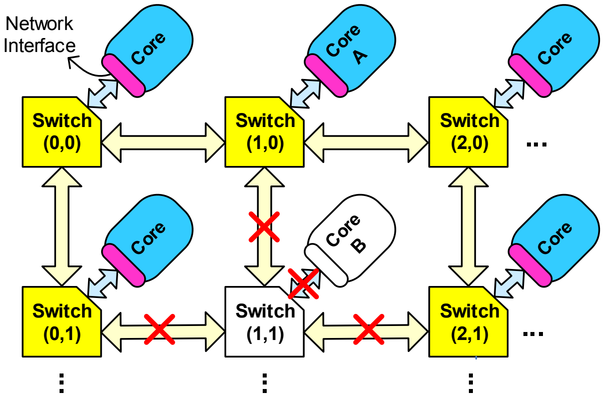

Figure 1: Port and link loss

In general, these faults can be mapped to a faulty port of the involved switch (Figure 2).

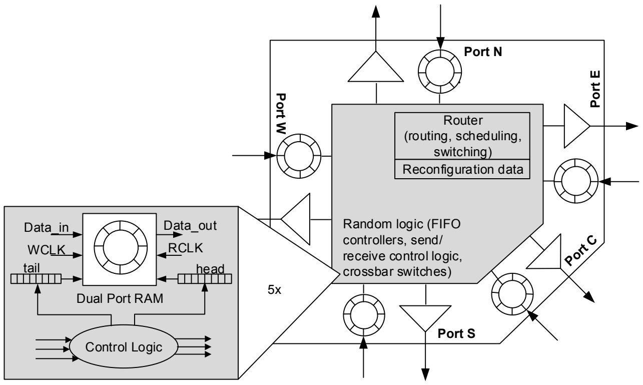

Figure 2: Generic switch structure

The usual way to deal with faulty switches or links at the physical layer is disabling the complete switch. Diagnosis may try to locate the fault with higher resolution, and to point to a faulty gate or line at gate level. Then, only ports have to be disabled which may be affected by this fault, and any remaining functionality can be reused further on. Overall, this leads to a higher perfomability and less degradation in the faulty case.

To a large part, the switch consists of combinational logic and some rather regular memory elements, for instance FIFOs and control logic. The regularity of the switch allows extracting a substantial part of its circuitry as combinational logic, which can be subject of further diagnosis.

3.1 Generalized Fault Modeling

For recent technologies, the stuck-at fault model reaches its limits and more expressive fault models are needed. The conditional line flip model [33] consists of a signal a at a certain fault site and a condition [cond] that activates the fault and is described by a Boolean, temporal or even random expression. For instance, $a \oplus [b \wedge \bar{a}]$ describes that $b = 1$ overwrites the 0 at a, the formula $a \oplus [\bar{a}_{-1} \wedge a]$ is a slow to rise transition fault, and $a \oplus [b_{-1} \wedge \bar{b} \wedge a]$ models crosstalk from b to a. This generalized fault model is able to describe both faults in the communication links or at ports, and faults in the gate level structure.

3.2 Topological preprocessing

A structural fault has only impact on those ports, which are topologically reachable from the fault site (see Figure 3). This straightforward observation already provides a good approximation of the set of functions, which are not affected and can still be used. However, most signals close to input ports have structural paths to many outputs and even to the router state. Not all of these paths can propagate the error signal. Hence, the topological preprocessing is pessimistic and further analysis techniques may obtain a better approximation of the intact functions.

3.3 Functional reasoning

Functional reasoning determines exactly the switch functions affected by a fault and the corresponding parts to be disabled. An appropriate method for this is provided by combinational, constrained ATPG. A fault *f* is injected, and the control inputs are constrained in such a way that it can only be detected at a certain subset of outputs.

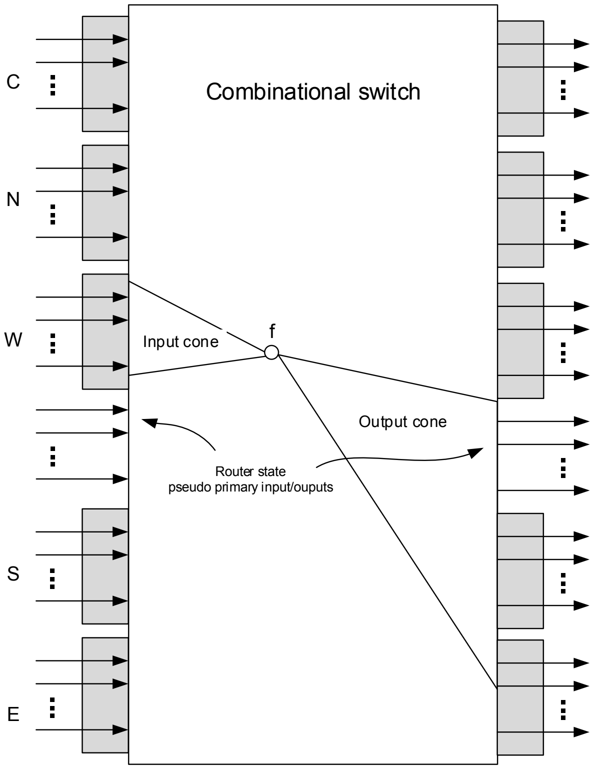

Figure 3: Combinational switch representation

Complete structural fault coverage for the conditional line flip model with SBST patterns poses a challenge, which is tackled by innovative ATPG techniques. The SBST pattern generation is modeled as a Boolean satisfiability (SAT) problem in conjunctive normal form (CNF). The SAT instance has to model three aspects:

- *Circuit Model:* The combinational logic and interconnect of the SUT is described in CNF using the Tseitin transformation. The result is a set of clauses Φ_C describing the combinational part of the switch.
- *Fault Injection:* The Conditional Line Flip (CLF) calculus described above is used as a generalized fault model to describe arbitrary defect mechanisms in the switch logic and the links. In order to model a fault at the location *f* the downstream logic has to be duplicated, and the fault has to be modeled by $f \oplus [cond]$. The expression *cond* is a free variable to guarantee the detection of any functional misbehavior. The set of clauses Φ_C^f describes now fault free and possible erroneous signals.
- *Output propagation:* For each output port and for each signal line to the router control logic it is checked whether a fault can be propagated. This is achieved by searching a satisfying assignment of Φ_C^f which also leads to a function mismatch at one of the outputs to be checked.

If ATPG fails to propagate the error condition to these outputs, it is proven that certain switch functions are not affected by this fault. Otherwise, three cases have to be distinguished:

- The switch has to be disabled completely, if
  - the faulty behavior cannot be explained by a single fault site *f*;
  - the error signal can propagate to router states

- An output port has to be disabled, if it is not the specified target of the switch function, but an error signal can be propagated to this port.
- A switch function ($North \rhd East$, e.g.) has to be disabled, if ATPG can propagate the error signal along this functional path.

Since the complexity of the combinational parts of a single switch is rather moderate, the technique is rather efficient despite the repeated call of ATPG.

4. DATA LINK LAYER

In an embedded switch, the test data developed above cannot be directly applied, but has to follow the network format. For microprocessors, the benefits of structural testing and functional testing are combined by a so-called structural software-based self-test (SBST) [6][7][24][19][35]. In this technique, ATPG provides deterministic, structural test patterns, which are transformed into arguments of a sequence of valid instructions. In a similar way, as functional test of switches and links requires to transform deterministic test patterns into valid packets of an NoC, this approach can be considered as a structural software-based self-test (SBST) scheme for Networks-on-Chip. Structural faults in NoC switches and interconnects are targeted and tested by valid NoC packets without the need for dedicated test infrastructure. Such an SBST scheme combines the advantages of state-of-the-art structural and functional test approaches for NoC infrastructure.

Figure 4 illustrates the principle of SBST in the scope of NoCs. As an example, in the mesh topology, every switch is connected to four neighboring switches and a Processing Element (PE) is attached to each switch. The Switch Under Test (SUT) is tested by applying a set of test patterns to its incoming links and by observing the test responses at the outgoing links. The test patterns form valid NoC packets, and do not require putting the system in a non-functional test mode. Here, we assume that the set of test packets is generated by software running on the processing elements (PE) attached to the NoC.

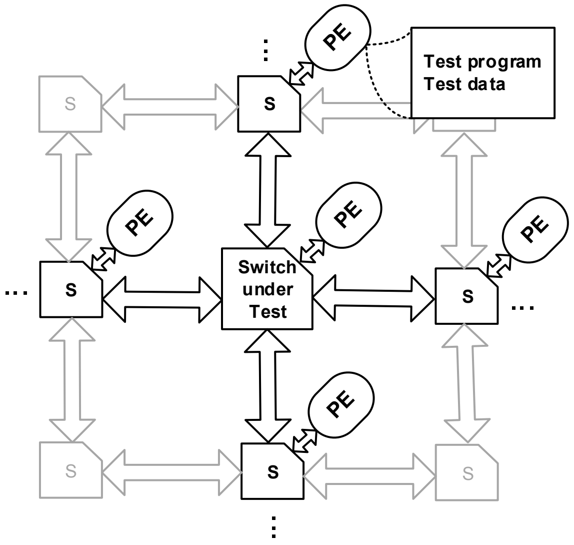

Figure 4: SBST for NoCs

The generated test packets target structural faults in the SUT and its links under the single fault assumption. The resulting test responses are captured and evaluated by the test programs in the adjacent PEs. The SBST starts when all PEs surrounding the SUT have sufficient resources to run the test program. A local signal (such as the Ack/Req. signal used for link flow control) can be utilized to synchronize the launch of the test programs running on the PEs involved in testing a SUT. The switches and PEs give the highest priority to test packets and bypass their caches.

Since the switches are identical, the SUT access time through all the incoming links is deterministic. Moreover, once the test begins, normal packets are not routed through the SUT. The complete NoC is tested by consecutively testing all contained switches. Depending on the network topology and the switch location, the SBST pattern generation is adjusted such that only available neighboring PEs contribute in testing. For example, in a 2D mesh a switch at the boundary has three neighbors, consequently its test patterns contain input values for only three input ports of the switch.

The key concept of SBST for NoC switches is the generation of efficient test patterns that achieve high fault coverage. In contrast to scan-based testing, direct controllability and observability of the sequential states of the switch (i.e. pseudo primary inputs and outputs) is not possible, and the sequential behavior of the switch has to be modeled as well. For this purpose, one can apply time-frame expansion to the combinational circuit Φ_C and obtain a sequential model Φ_s^T, where T denotes the necessary number of time steps as depicted in Figure 5.

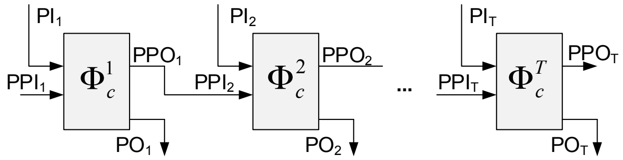

Figure 5: Unrolled switch

Also, fault injection becomes more difficult compared to the techniques discussed above, since a structural fault has to be modeled in all the T different time steps leading to a more complex set of clauses $\Phi_{SF}^{f,T}$ for the faulty instance.

In addition, only valid packets can be accepted as test patterns in order to utilize the packet based communication platform of the NoC for SBST. For this purpose, we create a function $f_{in}(i)$ denoting a Boolean formula that defines the functional input constraints. Essentially, this formula describes a data sequence, which is a well-formed package as seen in Figure 6.

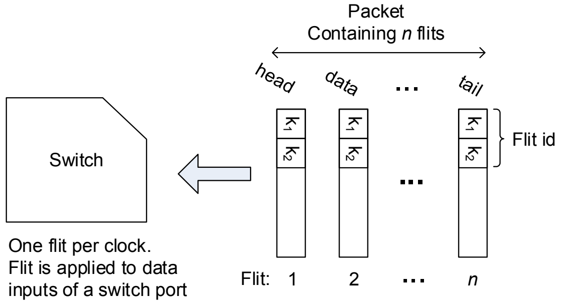

Figure 6: Packet format to be encoded by clauses

A functional test input has to satisfy now the formulas $\Phi_{SF}^{f,T}$, Φ_s^T, and $f_{in}(i)$ and has to enforce at least one output variable in $\Phi_{SF}^{f,T}$ being different from the corresponding one in Φ_s^T. The test set generated this way is valid for all regular switches and has to be stored only once.

5. NETWORK LAYER

5.1 Fault Classification

On the network layer, the direct correspondence to structural faults is lost and has to be reconstructed in order to evaluate the fault coverage of a test procedure and to locate faults with sufficient resolution. For this reason, the satisfiability-based (SAT) approach outlined above is used to classify structural faults into functional failure classes. Fault classification is especially useful to extend the functional failure classes, so that the structural fault coverage of the corresponding functional test increases. It determines which structural faults cause a certain functional failure. Besides, it provides a weighted functional failure classification with respect to the number of structural faults in each class.

The method includes four tasks:

1) Definition of functionalities of an NoC switch, and formalization of the corresponding failure modes.
2) Mapping the failure modes to the switch structure in the form of clauses to allow test generation by modern satisfiability solvers.
3) Modeling structural faults by clauses and adding these clauses to the failure mode description.
4) Solving the SAT problem allows now to generate data input for the functional test and to quantify the structural faults covered by each of the functional failure modes.

The outcome of this method consists of functional data packets for the switches and links, which can be applied in system mode and form highly effective test sequences. The experimental results show that functional tests generated this way achieve significantly higher fault coverage than the ones obtained by commercial sequential ATPG tools [7].

5.2 Functional Failure Modes for NoC Switches

The correlation of structural faults to high level faults of an NoC has a key role in the success of a functional test method. For this reason, functional failure modes must be carefully defined. This subsection describes some important failure modes.

The specification of an NoC switch implies the following functionalities:

- The received data is routed via the correct output port.
- The data is left intact.
- No data is lost.
- No new data is generated.

Accordingly, the functional failure modes of an NoC switch are defined as:

- Misrouting: The received packet is routed to the wrong output port. This fault may cause deadlock in the network.
- Data corruption: The data is corrupted for at least one flit in the packet.
- Packet/flit loss: At least one flit of the received packet is never delivered to the output port of the switch.
- Garbage packet/flit: A new packet/flit is generated and routed to the output port. This includes routing a received packet to more than one output port, or generating spurious flits among the flits of a packet.

5.3 Method Overview

In order to classify and weight the failure modes and additionally generate the corresponding functional test, models have to be generated which include the fault free switch, the faulty instance and the functional failure modes. The clause sets have been described in the sections above. This section concentrates on the description of the functional failure by a set of clauses.

Figure 7 depicts the switch interfaces, which have to be used to model a functional failure mode. By using the signals of the interface, any of the four failure modes can be expressed as a Boolean formula Φ_{FF}^{T}.

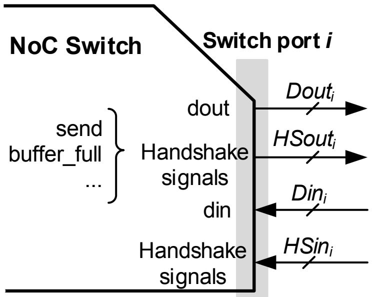

Figure 7: Interface signals for modeling failure modes

The SAT instance Φ_R explaining the relation between the target functional failure and the structural faults is built using the definition of the functional failure and the good and faulty copy of the switch:

$$\Phi_R = \Phi_{FF}^{T} \wedge \Phi_{s}^{T} \wedge \Phi_{SF}^{f,T}.$$

This formula is used for fault classification in order to extract the relation between low-level structural faults and the defined functional failure classes.

The classification results can be used to find an appropriate fault tolerant technique for the NoC switch. For more probable functional failure modes, a faster fault tolerance technique is preferred.

Checking for functional failures can be done either switch-to-switch or end-to-end. Detecting a functional failure in a switch-to-switch manner requires additional hardware and increases the component's latency. Nevertheless, an end-to-end retransmission introduces a higher performance penalty in case of an error. The classification does not only quantify the structural faults in the functional failure classes, but also determines which structural fault locations cause certain functional failures. This information can be used to make a cost-aware fault tolerant decision at multiple abstraction levels.

6. SUMMARY AND FUTURE WORK

Test, diagnosis and fault tolerance techniques are available on the different layers of the network, but they have to date largely been applied in isolation. Possible interactions between these layers have to be described and investigated in order to optimize the tradeoff between hardware and timing overhead for test and diagnosis on the one hand and the fault efficiency on the other hand.

Future work has to model and implement automated multi-layer interaction with respect to concrete NoC topologies and routing policies. An actual NoC design incorporating cross-layer test, diagnosis, and eventually fault tolerance appears to be a still distant future.

7. ACKNOWLEDGMENTS

This work has been supported by the German Research Foundation (Deutsche Forschungsgemeinschaft - DFG) under grant WU 245/12-1 and RA 1889/4-1 (project ROCK).

8. REFERENCES

[1] Abdel-Khalek, R. and Bertacco, V. 2012. Functional post-silicon diagnosis and debug for networks-on-chip. In *Proc. IEEE/ACM Int'l Conf. on Computer-Aided Design* (ICCAD), 557-563.

[2] Aisopos, K., Chen, C. H. O., and Peh, L. S. 2011. Enabling system-level modeling of variation-induced faults in networks-on-chips. In *Proc. 48th Design Automation Conference* (DAC), 930-935.

[3] Amory, A., Briao, E., Cota, E., Lubaszewski, M., and Moraes, F. 2005. A scalable test strategy for network-on-chip routers. In *Proc. of IEEE Int'l Test Conf.* (ITC), 590–599.

[4] Bertozzi, D., Benini, L., and De Micheli, G. 2005. Error control schemes for on-chip communication links: the energy-reliability tradeoff. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 24, 6, 818-831.

[5] Bushnell, M. and Agrawal, V.D. 2000. *Essentials of electronic testing for digital, memory and mixed-signal VLSI circuits.* Chapters 7 & 8. Springer Science & Business Media.

[6] Chen, L. and Dey, S. 2001. Software-based Self-testing Methodology for Processor Cores. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 20, 3, 369–380.

[7] Corno, F., Cumani, G., Sonza Reorda, M., and Squillero, G. 2003. Fully automatic test program generation for microprocessor cores. In *Proc. Design, Automation and Test in Europe* (DATE), 1006–1011.

[8] Dalirsani, A., Hatami, N., Imhof, M.E., Eggenberger, M., Schley, G., Radetzki, M., Wunderlich, H.-J. 2014. On Covering Structural Defects in NoCs by Functional Tests. In *Proc. 23rd IEEE Asian Test Symposium* (ATS), 87-92.

[9] Dalirsani, A., Holst, S., Elm, M., and Wunderlich, H.-J. 2011. Structural test and diagnosis for graceful degradation of NoC switches. *Journal of Eletronic Testing: Theory and Applications 28, 6*, 831–841.

[10] Dalirsani, A., Imhof, M.E., and Wunderlich, H.-J. 2014. Structural Software-Based Self-Test of Network-on-Chip. In *Proc. 32nd IEEE VLSI Test Symposium* (VTS), 1-6.

[11] Dalirsani, A., Kochte, M.A., Wunderlich, H.-J. 2014. Area-Efficient Synthesis of Fault-Secure NoC Switches. In *Proc. 20th IEEE Int'l On-Line Testing Symposium* (IOLTS), 13-18.

[12] Garbade, A., Weis, S., Schlingmann, S., Fechner, B., and Ungerer, T. 2013. Fault localization in NoCs exploiting periodic heartbeat messages in a manycore environment. In *Proc. 27th IEEE Int'l Parallel and Distributed Processing Symp. Workshops & PhD Forum* (IPDPSW), 791–795.

[13] Grecu, C., Pande, P., Ivanov, A., and Saleh, R. 2006. BIST for network-on-chip interconnect infrastructures. In *Proc. 24th IEEE VLSI Test Symp.* (VTS), 1-6.

[14] Hosseinabady, M., Dalirsani, A., and Navabi, Z. 2007. Using the inter- and intra-switch regularity in NoC switch testing. In *Proc. Design, Automation & Test in Europe* (DATE), 1–6.

[15] Jantsch, A., Lauter, R., and Vitkowski, A. 2005. Power analysis of link level and end-to-end data protection in networks on chip. In *Proc. IEEE Int'l Symp. On Circuits and Systems* (ISCAS), 1770-1773.

[16] Kakoee, M., Bertacco, V., and Benini, L. 2014. At-speed distributed functional testing to detect logic and delay faults in NoCs. *IEEE Trans. on Computers* 63, 3, 703–717.

[17] Kaufmann, M. 2012. *Reliable Communication by Fault-Tolerant Multilayer Routing*. Master Thesis, University of Stuttgart.

[18] Kohler, A., Schley, G. and Radetzki, M. 2010. Fault tolerant network on chip switching with graceful performance degradation. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 29, 6, 883-896.

[19] Kranitis, N., Paschalis, A., Gizopoulos, D., and Xenoulis, G. 2005. Software-Based Self-Testing of Embedded Processors. *IEEE Transactions on Computers*, 54, 4, 461–475.

[20] Lehtonen, T., Liljeberg, P., and Plosila, J. 2007. Analysis of forward error correction methods for nanoscale networks-on-chip. In *Proc. 2nd Int'l Conf. on Nano-Networks* (NanoNet), 1-5.

[21] Lehtonen, T., Wolpert, D., Liljeberg, P., Plosila, J., and Ampadu, P. 2010. Self-adaptive system for addressing permanent errors in on-chip interconnects. *IEEE Trans. on Very Large Scale Integration Systems*,18, 4, 527–540.

[22] Li, M., Jone, W.-B., and Zeng, Q.-A. 2006. An efficient wrapper scan chain configuration method for network-on-chip testing. In *Proc. IEEE Computer Society Annual Symp. on Emerging VLSI Technologies and Architectures* (ISVLSI), 147–152.

[23] Liu, C., Zhang, L., Han, Y., and Li, X. 2011. A resilient on-chip router design through data path salvaging. In *Proc. 16th IEEE Asia and South Pacific Design Automation Conf.* (ASP-DAC), 437–442.

[24] Paschalis, A. and Gizopoulos, D. 2005. Effective Software-Based Self-Test Strategies for On-line Periodic Testing of Embedded Processors. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24, 1, 88–99.

[25] Radetzki, M., Feng, C., Zhao, X., and Jantsch, A. 2013. Methods for Fault Tolerance in Networks on Chip. *ACM Computing Surveys* 46, 1, article 8.

[26] Raik, J., Ubar, R., and Govind, V. 2007. Test configurations for diagnosing faulty links in noc switches. In *Proc. 12th IEEE European Test Symposium* (ETS), 29-34.

[27] Rodrigo, S., Flich, J. Roca, A., Medardoni, S., Bertozzi, D., Camacho, J. Silla, F., and Duato, J. 2010. Addressing manufacturing challenges with cost-efficient fault tolerant routing. In *Proc. 4th ACM/IEEE Int'l Symp. on Networks-on-Chip* (NOCS), 25–32.

[28] Schley, G., Batzolis, N., Radetzki, M. 2013. Fault Localizing End-to-End Flow Control Protocol for Networks-on-Chip. In *Proc. 21st EUROMICRO Conference on Parallel, Distributed and Network-Based Processing* (PDP), 454-461.

[29] Schley, G., Radetzki, M. 2015. Fault Tolerant Routing for Hierarchically Organized Networks-on-Chip. In *Proc. 23rd EUROMICRO International Conference on Parallel, Distributed and Network-based Processing* (PDP), 379-386.

[30] Shamshiri, S., Ghofrani, A., and Cheng, K.-T. 2011. End-to-end error correction and online diagnosis for on-chip networks. In *Proc. Int'l Test Conference* (ITC), 1-10.

[31] Vitkovskiy, A., Soteriou, V. and Nicopoulos, C. A. 2012. Dynamically adjusting gracefully degrading link-level fault-tolerant mechanism for NoCs. IEEE *Transactions on Computer-Aided Design of Integrated Circuits and Systems* 31, 8, 1235-1248.

[32] Williams, M. and Angell, J. 1973. Enhancing testability of large-scale integrated circuits via test points and additional logic. *IEEE Trans. On Computers* 15, 5, 46–60.

[33] Wunderlich, H.-J. and Holst, S. 2010. Generalized Fault Modeling for Logic Diagnosis. In Wunderlich, H.-J.(Ed.), *Models in Hardware Testing*, ISBN: 978-90-481-3281-2, Springer-Verlag Heidelberg, 133-155.

[34] Zhang, Z., Refauvelet, D., Greiner, A., Benabdenbi, M., and Pecheux, F. 2011. Localization of damaged resources in NoC based shared-memory MP2SOC, using a distributed cooperative configuration infrastructure. In *Proc. 29th IEEE VLSI Test Symp*. (VTS), 229–234.

[35] Zhou, J. and Wunderlich, H.-J. 2006. Software-based self-test of processors under power constraints. In *Proc. Design, Automation and Test in Europe* (DATE), 430–435.