In-Field NoC-Based SoC Testing with Distributed Test Vector Storage

Jason D. Lee and Rabi N. Mahapatra
Texas A&M University
{jdlee,rabi}@cs.tamu.edu

***Abstract*—The operational lifetimes of SoC and microprocessors face growing threats from technology scaling and increasing device temperature and power density. In-field (or on-line) testing of NoC-based SoC is an important technique in ensuring system integrity throughout this potentially shorter lifetime. Whether in-field testing is conducted concurrently with normal applications or executed in isolation, *application intrusion* must be minimized in order to maintain system availability. Specialized infrastructure IP have been proposed to manage on-line testing by scheduling tests and delivering test vectors to the various cores within the SoC from a centralized location.**

However, as the number of cores integrated into a single chip continues to increase, issuing test vectors from a centralized location is not a scalable solution. These increased distances that test vectors must travel have become a major concern for on-line testing because of its direct impact on application intrusion in terms of energy consumption, network load, and latency. In this paper, we apply a distributed storage technique to bound and minimize this distance, thereby minimizing network load, energy consumption, and test delivery latency across the entire network. Our experiments show that test delivery latency and energy consumption is reduced by approximately 90% for moderately sized NoC.

I. Introduction

Multi-core systems-on-chip (SoC) using on-chip busses have become standard architectures for embedded systems as designers seek higher performance despite strict power limitations and time-to-market pressures. As integration continues to increase, the bus-based communication amongst the many cores becomes the primary system bottleneck. To address this concern, multi-hop networks-on-chip (NoC) were proposed as an alternative communication structure offering scalability, higher utilization, and re-usability [1], [2]. First generation NoC-based SoC have already been developed and produced for the server and embedded systems markets. Examples of these systems include Tilera's TILE64 [3], based on MIT's RAW architecture [4], and Intel's Terascale SoC [5].

In order to guarantee system integrity during operation, on-line (or in-field) testing strategies have been developed by various researchers [6]-[9]. The most prominent constraint of on-line testing, when compared to standard manufacturer testing, is the absence of external automatic test equipment (ATE) connected to the SoC which manages test vector delivery, scheduling, and results comparison. In this absence, test vectors and additional design for test (DFT) structures must be present on-chip.

On-line IP core testing, whether done concurrently with applications or as an isolated process, is an expensive task in terms of power consumption and performance degradation [8]. Performance degradation is especially critical during concurrent testing when the applications are subject to real-time constraints.

To further complicate this problem, current research has shown that as on-chip networks grow in size, power consumption and latency related to delivering test vectors across the chip becomes prohibitive when test vectors are stored in a central location [6]. In order to mitigate these constraints, storing test vectors in multiple locations within a single chip has been proposed; however, there has been no formal analysis on how best to distribute these tests.

On-line testing must minimize intrusion into the running applications. We consider intrusion to be the additional network traffic, energy consumption, and core test time. Each of these burdens is directly related to the distance that test vectors must travel across the network – the proximity of the test source to each core to be tested. The most extreme solution to minimizing the distance between test source and each core is to simply replicate the necessary test vectors across all cores in the network. Although network traffic overhead and its related energy costs are essentially reduced to zero, this is clearly not an attractive solution in regards to storage requirements.

This discussion leads us to the central question addressed by this paper. As the number of cores on a chip, and the on-chip networks connecting these cores, continue to grow, how should a system designer store and distribute test vectors within a SoC such that each core can access all test vectors within a bounded distance? By bounding the distance that test vectors must travel, on-line test intrusion can be controlled and estimated accurately, leading to a more reliable and available safety-critical system.

In this paper, we present a distributed test vector storage technique for safety-critical SoC using a torus-based on-chip network that can accommodate storage, power, latency, and availability constraints in an optimal fashion. The on-chip networks we study in this paper focus on the 2D-torus topology – a popular topology in research and in practice due to its efficient hardware implementation, small diameter,

978-1-4244-2658-4/08/$25.00 ©2008 IEEE

and simple routing. The main contributions of this paper are the following:

- *Proposes the use of a formalized, fully distributed storage technique* (t-interleaving on tori) *to bound the distance test vectors travel within the network, reducing on-line test intrusion*
- *Illustrates the importance of bounding the distances test vectors must travel within an SoC, and the design trade-offs between this* delivery distance *and* storage costs
- *Applying this scheme can reduce test delivery latency and energy consumption by approximately 90% and total network load by 76% for the moderately sized NoC simulated in our experiments*

The rest of this paper is organized as follows. Section 2 provides preliminary information of NoC-based SoCs, on-line testing, and distributed storage theory. Section 3 describes the experimental setup used to measure the performance of delivering test vectors using the standard single-source approach and the described distributed storage technique. Section 4 presents our analysis of distributed test vector storage in terms of network traffic load, test delivery latency, required energy consumption, and on-chip storage requirements for various SoC configurations. Finally, Section 5 summaries this work and describes future directions for research regarding this technique.

II. PRELIMINARIES

A. Networks-On-Chip

Networks-on-Chip (NoC) have been identified as effective communication infrastructures for many-core SoCs, replacing traditional bus-based communication that limits scalability, availability and performance [1,2]. Typical architectures consist of intellectual property (IP) cores embedded in network tiles that interface to the NoC via *core-network interfaces* (CNI). On-chip *routers* route communication between network tiles providing multi-hop communication between source and destinations. Fig. 1 illustrates the typical NoC architecture used in this research.

A variety of topologies for on-chip network were studied by researchers, including mesh, torus, ring, fat-tree, and circulant graphs [11], [12]. In this research, we focus solely on the popular 2D-torus topology. However, the techniques presented in this paper can be generalized to any regular network topology for which a distance metric is defined.

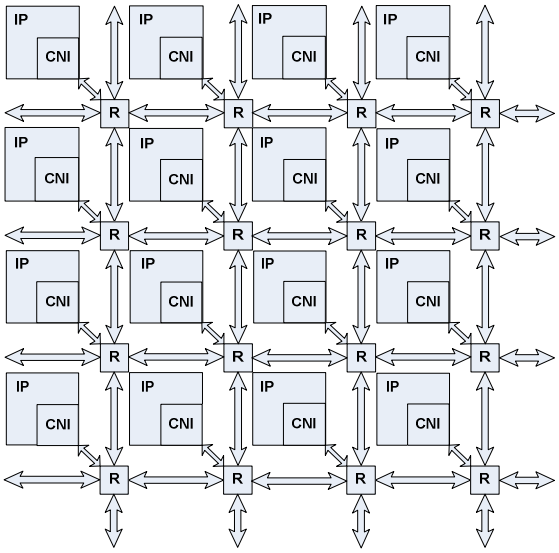

Fig.1. NoC Architecture with 2D 4x4 Torus Topology

B. In-Field Testing with NoC

In [10], the researchers analyzed the costs and benefits of reusing the NoC as a test access mechanism (TAM) for each core in the network. This was proposed as a response to the increasing difficulty of accessing deeply embedded cores within a SoC. It was shown that a NoC is indeed a feasible TAM due to minimal overheads; however, it was noted that test delivery times (and power consumption) depend on the distance between the test source and the core to be tested. Additionally, this variability in test delivery time may not be fully understood, since relatively small 5x7 and 4x8 2D-torus topologies were studied in that research.

Using the NoC as a test delivery infrastructure, researchers have proposed reusing infrastructure IP (I-IP) designed originally for manufacturer testing as a tool for on-line testing of the system [13],[14].

Taking this concept a step further, [7],[8] constructed a Test Infrastructure IP (TI-IP) capable of managing test scheduling, delivery, and intrusion for concurrent on-line test (COLT) of the SoC. COLT allows cores in the SoC to be tested in the presence of normally executing applications to maximize system availability. Due to the real-time constraints of these applications, COLT is extremely sensitive to application intrusion.

In a centralized test vector storage scheme, energy consumption, network load, and test delivery latency become dependent on the distance of the core under test to the TI-IP. Aside from the obvious scalability issues, this increases the complexity (and potentially decreases the accuracy) of calculating intrusion costs of on-line testing by the TI-IP. Ultimately, this may negatively affect the availability of the system. This motivates the need to bound test delivery distances for all cores in the SoC.

C. File Distribution in Torus Networks

This paper focuses on NoC based on the 2D-torus topology, also known as a k-ary 2-cube (Q_2^k). As the Hamming metric has been shown to efficiently describe relationships between nodes in hypercubes, the Lee metric has been shown to be a natural description of distances between nodes in tori [15].

The Lee metric for a 2D-torus of size n x n can be described as follows:

$$w_L(A) = \min(a_0, n - a_0) + \min(a_1, n - a_1)$$
$$d_L(A, B) = w_L(A - B) \quad (1)$$

Where w_L is the Lee weight of a node A within the 2D-torus, and a_0 and a_1 are the node's positions in the two dimensions. d_L is the Lee distance between two nodes in a torus, and like the Hamming distance, it is simply the weight of the difference between the two nodes. The Lee distance between two nodes can be intuitively described as the number of hops between two nodes in a torus topology. For a more in-depth discussion of the torus topology and their properties, please refer to [15,16].

Understanding the role of Lee metrics in tori is important when considering how to optimally distribute a file across a 2D-torus topology. Any test vector distribution scheme that bounds test vector delivery to each core must ensure that the complete set of test vectors are within a certain Lee distance to each core. We will now discuss how to distribute test vectors across a torus in more detail.

A file F (in our discussion, F is a set of test vectors) can be evenly divided into a set of file segments: $F = (F_0, F_1, \ldots, F_m)$. Each core can reconstruct the file F by retrieving the file segments from each of its neighbors within a radius *r*. It is necessary that for each core in the 2D-torus, all cores within that radius store a distinct file segment in order for the complete file F to be reconstructed.

As an example, Fig. 2 depicts an arbitrary core in a large 2D-torus which is storing file segment F_6 of 13 total file segments. The neighboring cores within the radius $r = 2$ store the remaining 12 file segments. When all 13 nodes are combined, a Lee sphere is created, and each node within that Lee sphere contains a distinct file segment. It can be clearly seen that the core at F_6 can access the entire file F by retrieving data from its neighbors within the radius *r*.

In order to guarantee every node in the 2D-torus can also access the entire file within the same radius, the Lee sphere centered at each node must contain nodes with distinct file segments. The file F must therefore be replicated across the entire torus in a spherical manner such that this constraint is satisfied.

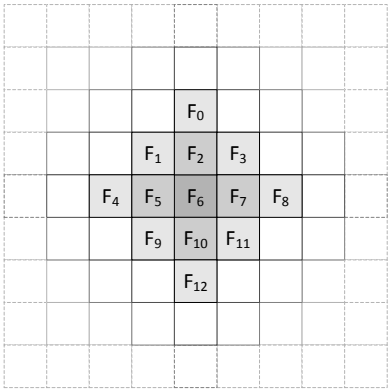

Fig. 2. A core at F_6 retrieving all file segments within radius 2

An alternative way of viewing this problem is to realize that replications of the same file segment F_i can be no closer than 2r + 1 hops apart.

In [17], this problem has been solved through *t-interleaving on tori*. t-interleaving on tori is formally defined as:

> *"Let G be a graph. By an* interleaving*, we will mean a vertex coloring, as follows. We say that G is* interleaved *(or there is an* interleaving *on G) if each vertex of G is assigned one of a finite number of distinct colors. We say that G is t-*interleaved *(or there is a t-*interleaving *on G) if every set of t vertices, forming a connected subgraph of G, is colored by t distinct colors."* [17]

In [17], $|S_t|$, the size of a Lee sphere with diameter *t*, has been proven to be the lower bound on how many file segments must be used to create a t-interleaving on a torus, and $|S_t|$ must divide *n* for a *n* x *n* 2D-torus to be a perfect, or optimal, t-interleaving. This paper studies perfect t-interleavings over tori; however, non-perfect t-interleavings can still achieve tight bounds on test delivery distances.

Looking back at Fig. 2, we see that the core storing file segment F_6 is at the center of a Lee sphere of radius r = 2, created by a 5-interleaving on that torus ($|S_5| = 13$).

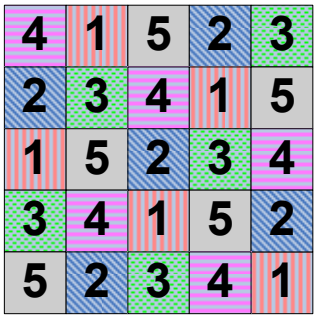

Fig. 3. 3-interleaving on 5x5 2D-torus

Fig. 3 depicts another example of t-interleaving on a smaller torus. Specifically, it depicts a perfect 3-interleaving on a 5x5 2D-torus. By observation, this example shows that any node within the graph can reach all colors within 1 hop. It also shows that nodes of the same color (or number label) are no closer than 3 hops apart. Another way to view this interleaving is to see that every node is the center of a Lee sphere of radius 1, and each Lee sphere contains 5 nodes with distinct colors (or numbers).

By using t-interleaving on tori for on-chip test vector storage, it is proven that any core in the network can access the complete set of test vectors within a defined radius. Bounding this test delivery radius for each core not only guarantees scalability, but on-line test intrusion can be much more easily estimated. The level of intrusion for each core no longer depends on proximity to the test source, simplifying scheduling decisions for on-line testing.

D. Systems with Heterogeneous Cores

It is apparent that the advantages of using this storage technique are maximized when all cores in the system are identical, or homogeneous, and the advantages are minimized when all the cores are different, or heterogeneous. This due to the fact that only a single set of test vectors is stored across the entire system, but multiple sets of test vectors may be required to test a multitude of varying cores.

This is a well-known problem in circuit testing, and various solutions have been proposed in research over the past decade. These solutions, such as broadcast scan [18] and Illinois scan [19], use a technique called *pattern sharing* which exploits the fact that deterministic test vectors are mainly composed of "don't care" bits. In fact, only 1% to 5% of the bits in a typical test vector are specified [20]. Therefore, differing test vector sets for heterogeneous cores can be combined into a single test vector set.

III. Experimental Setup

To measure the performance of using this distributed test vector technique, we simulate a NoC-based SoC using the NoCSim on-chip network simulator [21]. NoCSim is a SystemC cycle-accurate simulator which models IP cores, on-chip routers, CNI, and network links for any network topology to form a complete system.

The simulated on-chip network utilizes 64-bit links, and each IP core can transmit information in 64-byte packets. The test vector sets used in all experiments are 256KB in size; therefore, 4096 packets of information are transmitted from test source to test sink. The measurement of energy consumption is based on energy models developed in [22] which considers a system operating at 1GHz using 180nm technology.

Four systems are constructed for our experiments: a 25-node SoC in a 5x5 torus topology, a 64-node SoC in a 8x8 torus topology, a 100-node SoC in a 10x10 torus topology, and a 169-node SoC in a 13x13 torus topology. For each system, we simulate the behavior of delivering test vectors to each node in the system using a standard single-source approach and the previously described distributed technique based on t-interleaving.

For the 5x5 and 10x10 torus systems, 3-interleaving is used to distribute the test vectors across the network; the test vector set is split into 5 pieces, and each node can access all test vectors within 1 hop. For the 8x8 torus, 4-interleaving is used, therefore the test vector set is split into 8 pieces, and each node can access all test vectors within 2 hops. Finally, the 13x13 torus system uses 5-interleaving to distribute the test vectors. In this scheme, the test vector set is split into 13 pieces, and each node can access all test vectors within 2 hops.

IV. Performance Analysis

For this analysis, we calculate network load and measure test delivery latency and energy consumption assuming a test vector volume of 256KB. The actual test volume is not the focus of this research and it is only used for illustrative purposes.

A. Network Load

For evaluating total network load of test vector delivery on a 2D-torus NoC-based SoC, we use equations developed in [23] to analyze network requirements. Separate equations were derived for single-source test vector delivery and distributed test vector delivery.

Using these equations, Fig. 4 shows the reduction in total network load when using t-interleaving compared to single source test storage. For the 5x5 and 10x10 tori, 3-interleaving was used, whereas 4-interleaving and 5-interleaving was used for the 8x8 and 13x13 tori, respectively. The greatest network load reduction of 76% can be observed for the 13x13 torus. Greater network load reductions can be achieved for larger tori due to the ever increasing test vector delivery distances of the centralized scheme.

B. Latency

Test delivery latency was measured as the time from the IP core test source issuing the first test vector packet to the time the IP core test sink receives the last test vector packet.

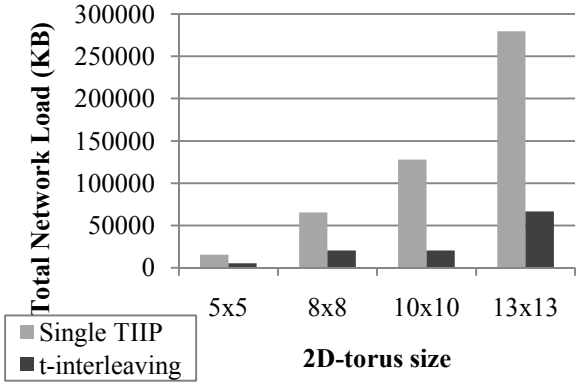

Fig. 4. Network load comparison as network size increases

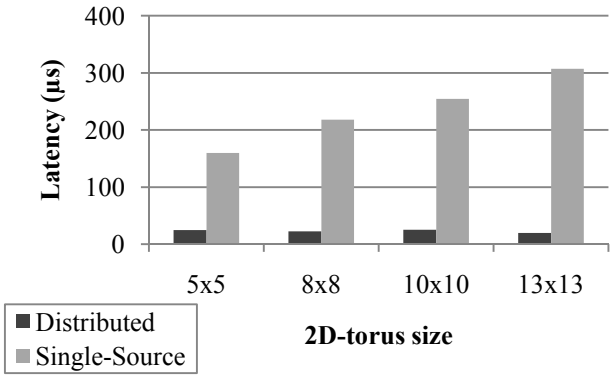

Fig. 5. Effect of NoC size on average test delivery latency

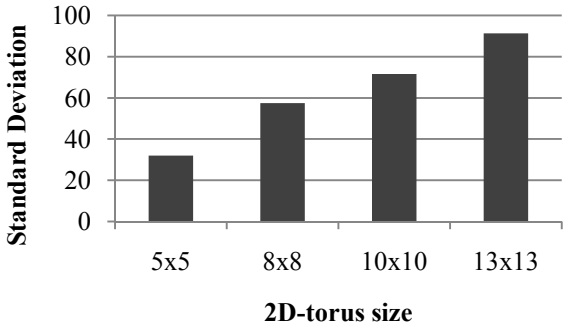

Fig. 6. Effect of NoC size on single-source test delivery latency deviation

Fig. 5 illustrates the effect of NoC size on average test vector delivery latency for both the single-source method and the distributed storage technique. It is clear that average test vector delivery latency is directly affected by network size for the single-source method; average test delivery latency increases from 159µs for the 5x5 torus to 307µs for the 13x13 torus. For safety-critical systems that must test, diagnose, and repair faulty components within milliseconds, this large delay in delivering test vectors may be unacceptable.

Fig. 5 also shows that test vector delivery latency is not sensitive to network size when using distributed test vector storage. For all network sizes simulated, test delivery latency is approximately 20µs. This translates into a latency reduction of 85% for the 5x5 torus to a 94% latency reduction for the 13x13 torus.

This dramatic reduction in latency is attributed to two factors. The average distance test vectors travel is reduced when using the distributed storage technique. This confirms previous findings that distance directly affects latency. Additionally, test vectors are issued from multiple test sources; this exploits the parallelism and bandwidth available in a point-to-point on-chip network.

Fig. 6 depicts the growing variation in test delivery time as NoC size increases. Intuitively, this is caused by growing variation in distance between test source and sink as the network size increases. For instance, in a 5x5 torus network, the distance between test source and sink is between 1 hop and 4 hops. However, in a 13x13 torus network this distance could be between 1 hop and 12 hops. Since test delivery latency is directly related to the distance test vectors travel, the increasing variation in distance leads to an increasing variation in latency. Fig. 6 shows that the standard deviation in latency increases by 285% between the 5x5 and 13x13 tori.

This increasing variation in latency complicates any in-field testing strategy that uses centralized test storage. Since any decision to apply a test during operation is based on the application intrusion, a test controller must separately consider the location for each IP core to be tested. Additionally, this increasing variance in latency shows that test delivery time becomes increasingly uncertain as test delivery distance grows. This makes any decision to apply tests over long distances during operation a risky proposition for safety-critical applications.

Table 1. Average Test Delivery Energy Consumption (µJ)

2D-torus size	Single-Source	Distributed
5x5	1522.33	256.55
8x8	7533.08	804.00
10x10	16382.66	1646.50
13x13	40001.61	2703.00

C. *Energy Consumption*

The energy consumed by the on-chip network resources was also measured during simulation to assess the energy and power requirements of transmitting test vectors between cores on a chip.

Table 1 describes the energy reduction possible when test vectors are distributed across a chip. The smallest system simulated, the 5x5 torus, shows that the average energy required to transmit test vectors from source to destination is reduced by 83%. This reduction increases with torus size, and the largest torus simulated, the 13x13 torus, shows a 93% reduction in energy consumption. This large reduction in the energy required to deliver test vectors to IP cores across the chip is extremely important for low-power and energy conscious designs which are becoming increasingly prevalent in embedded and safety-critical systems.

D. *Scalability and Redundancy*

For an n x n 2D-torus, the amount of redundancy introduced into the network through perfect t-interleaving is described by the following equation:

$$r = \frac{n^2}{|S_t|} \tag{2}$$

This equation simply computes the number of Lee spheres embedded within the torus.

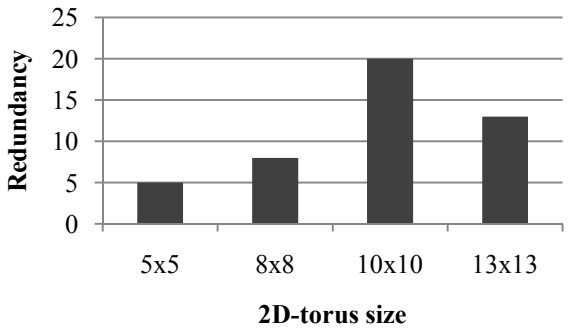

Fig. 7. Storage redundancy for various tori using distributed test vector storage

Fig. 7 shows the redundancy requirements for various t-interleaving schemes over different size 2D-tori using equation 2. In fact, for any *n* x *n* 2D-torus, the minimum required level of redundancy for a perfect t-interleaving is simply *n* since $|S_t|$ must divide *n*.

In Fig. 7, it is observed that the larger 13x13 torus requires less redundancy than the smaller 10x10 torus. This is due to the fact that 3-interleaving is used for the 10x10 torus (Lee spheres of size 5 are created), and 5-interleaving is used for the 13x13 torus (Lee spheres of size 13 are created). The 10x10 torus configuration contains 100 total nodes, and the test vectors are replicated across the 20 spheres created. The 13x13 torus configuration has 169 total nodes, and the test vectors are replicated across the 13 spheres created. Therefore, tests are replicated seven more times in the 10x10 torus configuration than the 13x13 torus configuration.

If less redundancy is desired, a designer may simply construct a sub-optimal t-interleaving over a torus. Another method to reduce redundancy requirements is to store file segments in only some nodes while still bounding test delivery distance. However, the main limitation of on-line testing noted in research literature has been test delivery distance and not storage.

V. Conclusions

In this paper, we have described using the t-interleaving technique to optimally distribute test vectors across a SoC to minimize the impact of in-field testing. By minimizing and bounding the distances test vectors must travel across a chip, system designers can better understand how testing will intrude into system applications. Additionally, test vector delivery latency and energy consumption is dramatically reduced, allowing for thorough in-field testing to be used in low-power and energy efficient systems.

Future directions for this research include studying various test scheduling techniques which account for this storage technique, further simulations using case studies of specific IP cores and their generated test vectors, and extending this technique to be used in topologies other than the 2D torus.

References

[1] L. Benini and G. De Micheli, "Networks on Chip: A New SoC Paradigm," *Trans. Computer*, Vol. 35, No. 1, 2002.
[2] W. Daly, and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," *Proc. Design Automation (DAC)*, 2001, pp. 684-689.
[3] Tilera Corporation. http://www.tilera.com
[4] M.B. Taylor, et al, "The Raw Microprocessor: A Computational Fabric for Software Circuits and General-Purpose Programs", *IEEE Micro*, Mar 2002, pp. 25-35.
[5] Intel Tera-scale Computing Research Program, http://www.intel.com/research/platform/terascale
[6] A. Manzone, et al, "Integrating BIST Techniques for On-line SoC Testing," Proc. *IEEE Intl. On-Line Testing Symposium (IOLTS)*, 2005, pp. 235-240.
[7] P. Bhojwani, and R. Mahapatra, "An Infrastructure IP for On-Line Testing of Network-on-Chip Based SoCs," *Proc. IEEE Intl. Symp. on Quality Electronic Devices,* 2007, pp. 867-872.
[8] P. Bhojwani, and R. Mahapatra, "A Robust Protocol for Concurrent On-Line Test (COLT) of NoC-based Systems-on-a-Chip," *Proc. Design Automation (DAC),* 2007, pp. 670-675.
[9] Y. Li, S. Makar, and S. Mitra, "CASP: Concurrent Autonomous Chip Self-Test Using Stored Test Patterns," *Proc. Design, Automation, and Test in Europe (DATE)*, 2008, pp. 885-890.
[10] E. Cota, L. Carro, F. Wagner, and M. Lubaszewski, "Power-Aware NoC Reuse of the Testing of Core-Based Systems," *Proc. Intl. Test Conference (ITC)*, 2003, pp. 612-621.
[11] J. Duato, et al, *Interconnection Networks: an Engineering Approach*, IEEE Computer Society Press, Los Alamitos, CA, 1997.
[12] C. Martinez, et al, "Dense Gaussian Networks: Suitable Topologies for On-chip Multiprocessors," *Intl. Journal of Parallel Programming*, Vol. 34, No. 3, 2006.
[13] Y. Zorian, "Guest Editor's Introduction: What is Infrastructure IP?," *IEEE Design & Test of Computers*, Vol. 19, 2002, pp. 3-5.
[14] L. Bolzani, et al, "Hybrid Soft Error Detection by means of Infrastructure IP cores," *Proc. Intl. On-Line Testing Symposium (IOLTS)*, 2004, pp. 79-84.
[15] B. Broeg, et al, "Lee Distance and Topological Properties of k-ary n-cubes," *Trans. Computer*, Vol. 44, No. 8, 1995.
[16] B. Broeg, et al. "Lee Distance, Gray Codes, and the Torus," *Telecommunication Systems*, Vol. 10, Nos. 1-2, 1998, pp. 21-32.
[17] A. Jiang, M. Cook, and J. Bruck, "Optimal Interleaving on Tori," *SIAM Journal of Discrete Math*, Vol. 20, No. 4, 2006, pp. 841-879.
[18] K. Lee, J. Chen, and C. Huang, "Broadcasting Test Patterns to Multiple Circuits," *IEEE Trans. on Computer Aided Design*, Vol. 18, No. 12, 1999, pp. 1793-1802.
[19] I. Hamzaoglu and J.H. Patel, "Reducing test application time for full scan embedded cores," *Proc. Intl. Symposium on Fault-Tolerant Computing*, 1999, pp. 260-267.
[20] L. T.Wang, C. E. Stroud, and N. A. Touba, *System-on-chip Test Architectures:* Nanometer Design for Testability. Morgan Kaufmann, Nov. 2007.
[21] NoCSim. http://codesign.cs.tamu.edu/nocsim
[22] Y. Jin, E. Kim, and K. Yum, "Peak Power Control for a QoS Capable On-chip network," *Proc. Intl. Conference on Parallel Processing (ICPP)*, 2005, pp. 585-592.
[23] J. D. Lee and R. Mahapatra, "Distributed Test Vector Storage for Safety-Critical NoC-based Systems", *Proc. IEEE Workshop on UCAS-4*, 2008, pp. 57-62.