Area Overhead and Test Time Co-Optimization through NoC Bandwidth Sharing

Fawnizu Azmadi Hussin, Tomokazu Yoneda, and Hideo Fujiwara
Graduate School of Information Science, Nara Institute of Science and Technology
Kansai Science City, 630-0192, Japan
{*fawniz-h, yoneda, fujiwara*}*@is.naist.jp*

Abstract

In this paper, a new approach to NoC test scheduling based on bandwidth-sharing is presented. The test scheduling is performed under the objective of co-optimizing the wrapper area overhead and the resulting test application time using two complementary NoC wrappers. Experimental results showed that the area overhead can be optimized (to an extent) without compromising the test application time. Compared to other NoC scheduling approaches based on dedicated paths, our bandwidth sharing approach can reduce the test application time by up to 75.4%.

1. Introduction

Several Network-on-Chip (NoC) architectures have been proposed such as Æthereal [1] and SoCIN [2]. A number of NoC scheduling methodologies [3–5] based on dedicated path approach have also been proposed. The use of NoC as a test access mechanism (TAM) relieves the need to add a conventional TAM [6] for test data transportation. However, dedicating a physical path between a tester and a core means that the path cannot be shared, thus preventing potential test concurrency. In addition, the path which passes through multiple store-and-forward routers does not guarantee jitter-free and timely data transportation. Hence, the standard IEEE 1500 [7] wrapper can not guarantee test data integrity.

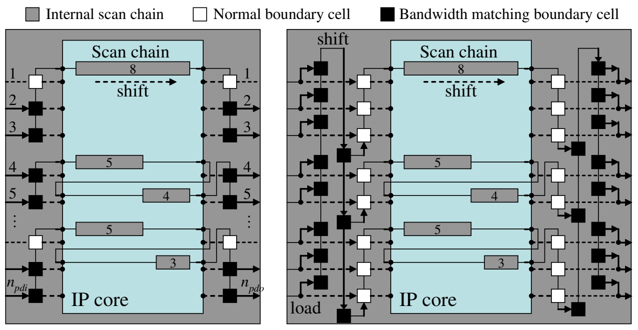
To overcome this shortcoming, the authors in [8] proposed an NoC wrapper which takes advantage of the guaranteed bandwidth and latency provided by the NoC to ensure test data integrity. While using the NoC as a TAM, the test data loading time of the NoC wrapper is comparable to the IEEE 1500 wrapper, which requires a more flexible but costly dedicated TAM, as implemented in [6]. However, the NoC wrapper requires much higher guaranteed bandwidth on the NoC than the actual rate of the test data loaded into the test wrapper. This is further explained in [9] in which two complementary wrapper architectures are proposed in order to overcome the limitations in [8].

In this paper, we propose an NoC scheduling mechanism which utilizes the two types of complementary NoC wrappers for area overhead and test application time co-optimization. The proposed approach also reuses the NoC and takes advantage of it's ability to allocate a specific amount of sustained bandwidth for any particular packet-based connection called a *virtual channel*, making it possible to divide a physical connection for multiple CUTs. The proposed bandwidth sharing achieves considerable reduction in test time, compared to the dedicated path approaches in [3–5].

2. NoC Wrapper Architecture

The IEEE 1500 [7] standard wrapper is designed to be used optimally when both the following conditions are true; (*i*) the TAM wires connected to a core can be assigned individually, and (*ii*) the timing of wrapper control signals can be controlled individually by an external ATE. When reusing the NoC in the functional mode as a TAM, the number of functional TAM wires is fixed. In addition, the ATE is unable to provide to each core directly the functional control signals during the test application. These restrictions render the standard 1500 wrapper unsuitable for the SoC testing based on the NoC-reuse. In [9], we have proposed two NoC wrappers to address these limitations.

The proposed test architecture, which uses the NoC's virtual channel, consists of two types of core wrappers. The Type 1 wrapper (Fig. 1(a)) requires minimal number of boundary scan cells, but wastes NoC bandwidth, except for some special configurations. The Type 2 wrapper (Fig. 1(b)) complements by means of the additional boundary cells. In this paper, we will explain a test scheduling methodology which utilizes both wrapper types in order to co-optimize the test time and the wrapper area costs.



(a) Type 1 NoC wrapper (b) Type 2 NoC wrapper

Figure 1. NoC-reuse wrapper architectures [9].

3. Test Scheduling through Bandwidth Sharing

The test strategy in this paper makes use of the NoC as a TAM. NoC is designed as an advanced SoC interconnect [1, 2] to provide a high bandwidth and modular infrastructure for on-chip communications. As such, the internal NoC bandwidth is typically much larger than the external I/O bandwidth. In this paper, we consider the test application of such SoCs utilizing an external tester as the test source/sink, which is interfaced through the low bandwidth I/O port. We will assume that a virtual channel can always be established from the I/O port to the target CUT as long as $\sum\{virtual\ channel\ bandwidth\} \leq \{I/O\ bandwidth\} \leq \{NoC\ internal\ bandwidth\}$. Under this assumption, the wrapper area and test time co-optimization problem addressed in this paper can be formulated as an I/O bandwidth distribution and core test scheduling problem as follows:

Ψ_S: Given an SoC C with M cores, a maximum I/O bandwidth, $B^{i/o}_{max}\ bps$, and a test frequency for all cores, f_m, where each core consists of n_i functional inputs, n_o functional outputs, n_b bidirectionals, k internal scan chains of length $l_1, l_2, ..., l_k$, for each core i determine
(1) the wrapper type and the allocated I/O bandwidth, $B_{scheduled}(i)$, for the test data transportation, and
(2) the starting time, $t_{start}(i)$, and end time, $t_{end}(i)$, of the test application
such that the total test application time and the area overhead are co-optimized under given priority weights α and β, respectively, where $\{\alpha, \beta\} \in [0, 1]$ and $\alpha + \beta = 1$.

Before explaining the schedule optimization algorithm (Sect. 3.3), we first clarify two required components of the algorithm in sections 3.1 and 3.2.

3.1 Wrapper Design Optimization

In order to achieve the objective (1) of Ψ_S, we first defined, in [9], the problems of optimizing the number of wrapper scan chains (n_{sc}) for both Type 1 and Type 2 wrappers under given constraints as follows:

Ψ_B: Given a core as in Ψ_S, and a maximum bandwidth for the virtual channel between the core and the ATE, $B^{vc}_{max}\ bps$, find the number of wrapper scan chains, n_{sc}, such that (*i*) the TAT is minimum, (*ii*) the required bandwidth, $B_{req} \leq B^{vc}_{max}$, and (*iii*) n_{sc} is minimum subject to objectives (*i*) and (*ii*).

Ψ_T: Given a core as in Ψ_S, and a maximum TAT, T_{max}, find the number of wrapper scan chains, n_{sc}, such that (*i*) the required bandwidth, B_{req}, is minimum, (*ii*) TAT $\leq T_{max}$, and (*iii*) n_{sc} is minimum subject to objectives (*i*) and (*ii*).

The TAT of a core is monotonically decreasing with regards to increasing number of wrapper scan chains. Therefore, the optimum solution to Ψ_B can be found in polynomial time, even with an exhaustive search. The solution is a Pareto-optimal point [6], where the corresponding wrapper configuration requires a sustained bandwidth, $B_{req} \leq B^{vc}_{max}$. A similar search algorithm was also implemented for problem Ψ_T in [9].

The area overhead for Type 1 and Type 2 wrappers can be estimated by the number of boundary cells given in equations (1) and (2), respectively. We decided not to include the wire routing cost because of its dependency on I/O placement, and to minimize the algorithm complexity. The extra $(+n_{pdi} + n_{pdo} + 2 \cdot n_{sc})$ in equation (2) are due to the additional input/output buffers (black squares in Fig. 1(b)) that perform bit-width matching. Equation (3) gives the relative cost of using a Type 2 instead of the Type 1 wrapper. Equation (4) gives the opposite cost.

$$H_{t1} = n_i + n_b + n_o \tag{1}$$
$$H_{t2} = n_i + n_b + n_o + n_{pdi} + n_{pdo} + 2 \cdot n_{sc} \tag{2}$$
$$Cost_{(t1 \rightarrow t2)} = \alpha \cdot \left(\frac{T_{t2} - T_{t1}}{T_{t1}} + \frac{B_{t2} - B_{t1}}{B_{t1}}\right) + \beta \cdot \frac{H_{t2} - H_{t1}}{H_{t1}} \tag{3}$$
$$Cost_{(t2 \rightarrow t1)} = \alpha \cdot \left(\frac{T_{t1} - T_{t2}}{T_{t2}} + \frac{B_{t1} - B_{t2}}{B_{t2}}\right) + \beta \cdot \frac{H_{t1} - H_{t2}}{H_{t2}} \tag{4}$$

For a given maximum bandwidth, B_{max}, the optimum configuration of a core i is determined by solving $\Psi_B(i, B_{max})$ to obtain the respective test application time (T_{t1} and T_{t2}) and required bandwidth (B_{t1} and B_{t2}) for the Type 1 and the Type 2 wrappers, respectively. If $Cost_{(t1 \rightarrow t2)} < Cost_{(t2 \rightarrow t1)}$, then the Type 2 wrapper is selected as a better wrapper configuration for the given B_{max}. Otherwise, the Type 1 wrapper is chosen. This cost function will be the basis for wrapper selection under given cost weights α and β.

3.2 Lower Bound on Test Time

The first lower bound is based on the dominant core effect. For each core i, assuming that it is given the maximum available bandwidth, $B^{i/o}_{max}$, its test time can be determined by $T(\Psi_B(i, B^{i/o}_{max}))$, which represents the TAT returned by Ψ_B search algorithm for Core i when the given maximum bandwidth is $B^{i/o}_{max}$. The TAT of an SoC C (equation (5)) cannot be shorter than the TAT of the longest core $i \in C$.

$$T^1_{LB} = max_{i \in C}\{T(\Psi_B(i, B^{i/o}_{max}))\} \tag{5}$$

For a bounded $B^{i/o}_{max}$, T^1_{LB} does not represent a meaningful lower bound. Therefore, a tighter lower bound based on the I/O capacity to transfer test vectors into the SoC is formulated as follows. The TAT of a core with one wrapper scan chain can be represented by equation (6) where $si = n_i + n_b + \sum_k l_k$, $so = n_o + n_b + \sum_k l_k$, and v_m is the number of test vectors. The second lower bound can be calculated as in equation (7), where f_m is the scan frequency for all cores. Equation (8) gives the overall lower bound.

$$T(i) = (max(si, so) + 1) \times v_m + min(si, so) \tag{6}$$
$$T^2_{LB} = \sum_{i \in C}\{T(i)\}\ /\ (B^{i/o}_{max}/f_m) \tag{7}$$
$$T_{LB} = max(T^1_{LB}, T^2_{LB}) \tag{8}$$

```
Procedure: OptimizeNoCSchedule ($C, B^{i/o}_{max}, v_{gain}, v_{bottleneck}$)
------------------------------------------------------------------------
Data Structure: Schedule
    $t_{start}(i)$; /*start time of Core $i$*/
    $t_{end}(i)$; /*end time of Core $i$*/
    $B_{scheduled}(i)$; /*allocated bandwidth for Core $i$*/
------------------------------------------------------------------------
 1. PreferredBandwidth ($C, B^{i/o}_{max}, v_{gain}, v_{bottleneck}$)
 2. $B_{free} \leftarrow B^{i/o}_{max}$; $t_{current} \leftarrow 0$;
 3. While $C \neq \emptyset${
 4.   If $B_{free} > 0${
 5.     If Core $i \in C$ can be found such that
          $B_{pref}(i) \leq B_{free}$ AND $T_{pref}(i)$ is maximum {
 6.       If ($C - \{i\} = \emptyset$), ScheduleLastCore ($i$);
 7.       Else, UpdateSchedule ($i, B_{pref}(i)$);}
 8.     Else {
 9.       Find $t_{next} \leftarrow t_{end}(i)$ such that
            $t_{end}(i) > t_{current}$ AND $t_{end}(i)$ is minimum;
10.       If Core $i$ can be found such that
            $(t_{current} + T(\Psi_B(i, B_{free})) \leq t_{next}$
            AND $B(\Psi_B(i, B_{free}))$ is maximum) {
11.         UpdateSchedule ($i, B_{free}$);}
12.       Else {
13.         DistributeFreeBandwidth ();
14.         $B_{idle} \leftarrow B_{free}$; $B_{free} \leftarrow 0$;}}}
15.   Else {
16.     Find $t_{next}$ as in Line 9;
17.     $t_{current} \leftarrow t_{next}$; $B_{free} \leftarrow B_{idle}$;
18.     For every Core $i$ such that $t_{end}(i) = t_{current}$ {
19.       $B_{free} \leftarrow B_{free} + B_{scheduled}(i)$;}}}
20. OptimizeMaxEndTime (Schedule);
21. Return Schedule;
```

Figure 2. Algorithm for solving Ψ_S.

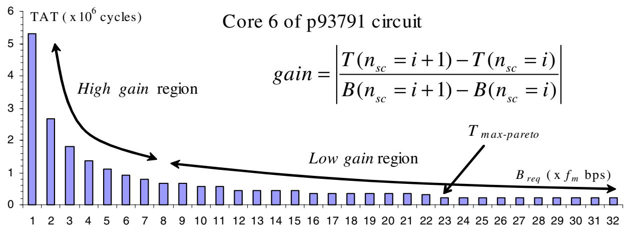


Figure 3. High (preferred) and low gain regions.

3.3 Scheduling through Rectangle Packing

We now introduce the concept of rectangles to represent core tests in the scheduling methodology based on NoC bandwidth sharing, which is inspired by the scheduling algorithm in [6]. The height of a rectangle represents the required bandwidth to obtain the test application time represented by the horizontal length.

The scheduling process (Fig. 2) starts with obtaining the *preferred bandwidth* for each core in the SoC. As illustrated in Fig. 3, the preferred bandwidth results after configuring the core wrapper with the number of scan chains in the "high gain" region. Gain represents the potential reduction in TAT of a core per unit of bandwidth allocated to that core. Therefore, it is better to assign additional bandwidth to a core that is still in the high gain region than one in the low gain region.

Figure. 4 describes the algorithm to determine the preferred

```
Procedure: PreferredBandwidth ($C, B^{i/o}_{max}, v_{gain}, v_{bottleneck}$)
------------------------------------------------------------------------
22. For each Core $i \in C${
23.   $T_{target1} \leftarrow T(\Psi_B(i, B^{i/o}_{max})) + [T(\Psi_T(i, \infty))$
          $- T(\Psi_B(i, B^{i/o}_{max}))] \times v_{gain}/100$;
24.   $T_{pref}(i) \leftarrow T(\Psi_T(i, T_{target1}))$;}
25. $avgT_{pref} \leftarrow$ Average $T_{pref}(i)$ for all $i \in C$;
26. $T_{LB} \leftarrow max(T^1_{LB}, T^2_{LB})$; /*lower bound, equation (8)*/
27. For each Core $i \in C${
28.   $T_{target2} \leftarrow min\{v_{bottleneck} \times avgT_{pref}, T_{LB}, T_{pref}(i)\}$;
29.   $B_{pref}(i) \leftarrow B(\Psi_T(i, T_{target2}))$;
30.   $T_{pref}(i) \leftarrow T(\Psi_T(i, T_{target2}))$;}
```

Figure 4. Calculating the preferred bandwidth.

```
Procedure: UpdateSchedule ($i, B_{given}$)
------------------------------------------------------------------------
31. $t_{start}(i) \leftarrow t_{current}$; $C \leftarrow C - \{i\}$;
32. $t_{end}(i) \leftarrow t_{current} + T(\Psi_B(i, B_{given}))$;
33. $B_{scheduled}(i) \leftarrow B(\Psi_B(i, B_{given}))$;
34. $B_{free} \leftarrow B_{free} - B_{scheduled}(i)$;
```

Figure 5. Scheduling Core i.

bandwidth for all cores. In line 23, a proper value of input percent v_{gain} shifts the target TAT from $T_{max-pareto}$ to the high gain region (Fig. 3). However, in some cases where the test application time is dominated by a large core such as Core 6 of p93791, selecting the high gain region for Core 6 would make it a bottleneck core, thus preventing further reduction of TAT. Therefore, in line 28 the variable $v_{bottleneck}$ together with the lower bound, T_{LB} (equation (8)), ensures that bottleneck cores are allocated larger preferred bandwidth, even in the low gain region.

When scheduling the last core (Line 6), the core start time and assigned bandwidth is chosen such that t_{end} is minimum. This is illustrated in Fig. 6(a) where three possible options are shown by the dotted rectangles. After all the cores are scheduled, in the final step (line 20), the current schedule of core i whose $t_{end}(i)$ is maximum, is reconsidered for further optimization. Without modifying the schedule for other cores, core i is rescheduled such that the new $t_{end}(i)$ is minimum (Fig. 6(b)). This process is repeated until no more reductions can be made to t_{end}.

4. Experimental Results

In this section, we present experimental results for several ITC'02 benchmark [10] circuits. From the design perspective, the cores whose $n_i + n_b < n_{pdi}$ or $n_o + n_b < n_{pdo}$ cannot be functionally interfaced to the NoC. As a result, two, four, and

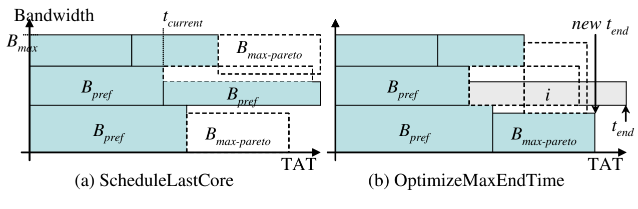


Figure 6. Further optimizing the schedule.

Table 1. Hardware-time co-optimization.

Cost weights		p93791noc $B^{i/o}_{max}$ = 6400 *Mbps* (T_{LB} = 435,039)			p22810noc $B^{i/o}_{max}$ = 6400 *Mbps* (T_{LB} = 102,965)			d695noc $B^{i/o}_{max}$ = 3200 *Mbps* (T_{LB} = 16,701)		
β	α	HOH	TAT	%/T_{LB}	HOH	TAT	%/T_{LB}	HOH	TAT	%/T_{LB}
0.00	1.00	9,303	464,252	6.7	5,768	122,091	18.6	2,396	17,827	6.7
0.25	0.75	8,653	464,252	6.7	5,680	122,280	18.8	2,300	17,827	6.7
0.50	0.50	7,673	471,175	8.3	5,698	122,280	18.8	2,300	17,827	6.7
0.75	0.25	7,673	471,175	8.3	5,412	130,591	26.8	2,110	18,184	8.9
1.00	0.00	6,557	483,411	11.1	3,810	134,466	30.6	1,676	18,494	10.7

Table 2. TAT for several $B^{i/o}_{max}$.

$B^{i/o}_{max}$ (*Mbps*)	p93791noc				p22810noc			
	HOH	TAT	T_{LB}	%/T_{LB}	HOH	TAT	T_{LB}	%/T_{LB}
3,200	8,849	923,842	870,079	6.2	5,584	232,816	203,015	14.7
6,400	9,303	464,252	435,039	6.7	5,768	122,091	102,965	18.6
9,600	9,009	347,378	290,026	19.8	5,798	102,965	102,965	0.0
12,800	8,885	235,285	227,978	3.2	5,798	102,965	102,965	0.0

five small cores are excluded from the modified benchmark circuits *d695noc*, *p93791noc*, and *p22810noc*, respectively, when $n_{pdi} = n_{pdo} = 32$. In addition, the optimum values (determined iteratively) of $v_{gain} \in [0..9]$ and $v_{bottleneck} \in [1..5]$ are used, with the scan frequency, $f_m = 100\ MHz$. The TAT reported in this paper is in number of scan clock cycles, where each cycle is equivalent to $1/f_m$ or $0.01\mu s$. The computation time is less than 10 seconds for the largest circuit.

In Table 1, the weights of hardware overhead cost (β) and TAT cost (α) are varied according to the constraints defined in Ψ_S. As β is increased, the total hardware overhead (columns labeled HOH) decreases while the test application time (columns labeled TAT) increases accordingly. This indicates that as we allow more hardware to be used, more bandwidth-efficient Type 2 wrappers can be used, allowing for a more efficient utilization of bandwidth, hence smaller "rectangles" to pack. Compared to the lower bound defined in Sect. 3.2, the TATs are on average 13% larger. The area overhead can be reduced considerably without affecting the TAT ($\beta = 0.0$ to 0.5) for all benchmark circuits. This happens when the Type 1 wrapper is used instead of the Type 2 wrapper for those cores that do not affect the overall TAT.

Table 2 shows the resulting HOH and TAT when B_{max} varies from 3.2 $Gbps$ to 12.8 $Gbps$, and $\alpha = 1, \beta = 0$. This illustrates that without increasing the area overhead, the TAT can be reduced given larger I/O bandwidth, $B^{i/o}_{max}$. This is typically the case because the functional I/O frequency is typically higher than the scan frequency. For the dedicated TAM based approach, TAT reduction can only be achieved by adding costly TAM wires.

Table 3 compares our bandwidth sharing approach with the dedicated path (DP) approaches [3–5]. In the DP approaches, a pair of NoC input and output ports can be used to test only one core at a time. Assuming that there is only one I/O port pair, the TAT for DP approach is the sum of each individual core test (sequential testing). Our approach enables parallelism through bandwidth sharing, which proves to be more efficient, with up to 75.4% TAT reduction.

Table 3. Test application time of dedicated path (DP) and shared bandwidth (SB) approaches.

Channel Bitwidth	d695noc			p93791noc			p22810noc		
	DP	SB	%red.	DP	SB	%red.	DP	SB	%red.
16	49,135	21,768	55.7	1,861,439	907,419	51.3	655,253	229,598	65.0
32	31,317	17,827	43.1	1,211,254	464,252	61.7	510,954	125,591	75.4

5. Conclusion

We have presented a new approach to NoC testing through bandwidth sharing, utilizing NoC-reuse wrappers. It was shown experimentally that it is not always necessary to use the expensive Type 2 wrappers in order to obtain a minimum TAT; the low-cost Type 1 wrappers can be used effectively without compromising the overall TAT. Compared to the previously published NoC test scheduling based on dedicated path approach, the proposed bandwidth sharing approach is much more efficient and flexible.

Acknowledgements

This work was supported in part by Japan Society for the Promotion of Science (JSPS) under Grants-in-Aid for Scientific Research B(No. 15300018) and for Young Scientists (B)(No.18700046). The authors would like to thank Prof. Michiko Inoue, Dr. Satoshi Ohtake and members of Computer Design and Test Laboratory in Nara Institute of Science and Technology for their valuable comments.

References

[1] A. Radulescu, J. Dielissen, S. G. Pestana, O. P. Gangwal, E. Rijpkema, P. Wielage, and K. Goossens, "An Efficient On-Chip NI Offering Guaranteed Services, Shared-Memory Abstraction, and Flexible Network Configuration", *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, Vol. 24(1), pp. 4-17, Jan. 2005.

[2] C. A. Zeferino and A. A. Susin, "SoCIN: A Parametric and Scalable Network-on-Chip", In Proc. 16^{th} Symposium on Integrated Circuits and Systems Design, 2003, pp. 169-174.

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

[4] A. M. Amory, E. Cota, M. Lubaszewski, and F. G. Moraes, "Reducing Test Time With Processor Reuse in Network-on-Chip Based Systems", In Proc. Integrated Circuits and Systems Design, 2004, pp. 111-116.

[5] C. Liu, Z. Link, and D.K. Pradhan, "Reuse-Based Test Access and Integrated Test Scheduling for Network-on-Chip", In Proc. Design, Automation and Test in Europe, 2006, pp. 303-308

[6] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "On Using Rectangle Packing for SoC Wrapper/TAM Co-Optimization", In Proc. IEEE VLSI Test Symposium, 2002, pp. 253-258.

[7] E. J. Marinissen, R. Kapur, M. Lousberg, T. McLaurin, M. Ricchetti, and Y. Zorian, "On IEEE P1500 standard for embedded core test", *Journal of Electronic Testing: Theory and Applications*, 2002, pp. 365-383.

[8] A. M. Amory, K. Goossens, E. J. Marinissen, M. Lubaszewski, and F. Moraes, "Wrapper Design for the Reuse of Networks-on-Chip as Test Access Mechanism", In Proc. IEEE European Test Symposium, 2006, pp. 213-218.

[9] F. A. Hussin, T. Yoneda, and H. Fujiwara, "Optimization of NoC Wrapper Design under Bandwidth and Test Time Constraints", IEEE European Test Symposium, 2007, pp. 35-40.

[10] E. J. Marinissen, V. Iyengar, and K. Chakrabarty, "A Set of Benchmarks For Modular Testing of SoCs", In Proc. International Test Conference, 2002, pp. 519-528.