Test Pin Count Reduction for NoC-based Test Delivery in Multicore SOCs*

Michael Richter
Department of Computer Science
University of Potsdam, 14482 Potsdam, Germany
miricht@cs.uni-potsdam.de

Krishnendu Chakrabarty
Dept. of Electrical and Computer Engineering
Duke University, Durham, NC 27708, USA
krish@ee.duke.edu

***Abstract*—We present the first pin-count-aware optimization approach for test data delivery over a network-on-chip (NoC). By co-optimizing core test scheduling and pin assignment to access points, the limited I/O resources provided by automated test equipment (ATE) can be used more effectively. This approach allows us to lower test cost by reducing test time for a given pin budget, or by reducing the number of test pins without impacting test time. To further improve resource utilization, we consider the use of MISRs for compacting the test responses of embedded cores. Experimental results for ITC'02 test benchmarks demonstrate that pin-count-aware co-optimization leads to shorter test times for a given pin-count budget and fewer pins for a given test-time budget. The results also highlight the advantages of the proposed use of output compaction.**

I. Introduction

It has been predicted that the number and complexity of cores in systems-on-chip (SOC) will continue to increase rapidly in the next few years [1]. Consequently, on-chip communication infrastructures will have to support higher bandwidth and numerous parallel data streams. In addition, reliability, delay and power issues will be more pronounced due to increasing wire lengths (relative to feature sizes). The ability of the current bus-based communication paradigm to cope with these problems is limited; moreover, maintaining a global synchronous clock for all cores of a large SOC is becoming increasingly difficult [1]–[4].

A packet-switched network-on-chip (NoC) is an emerging communication paradigm that promises to overcome the above limitations. The cores of an NoC-based SOCs can be tested using the NoC as the test-access mechanism, without the need for dedicated test-access mechanisms and their associated routing overhead.

Testing is typically carried out using automated test equipment (ATE), which provide test input data and evaluate the test responses. ATEs offer a limited number of tester channels that can be used to send data to or receive data from the device-under-test. Inefficient use of tester channels affects test cost in several ways. First, test-time targets might be missed, resulting in the need for more expensive ATEs that support more tester channels. Second, since the total available ATE memory is distributed among the different channels, test data volume may exceed tester memory capacity, necessitating time-consuming tester reloads, test set truncation (reducing test quality) or, again, use of more expensive ATEs. Even if test time cannot be reduced due to a bottleneck core, reducing the number of required pins might facilitate multi-site testing, i.e., allow us to test more SOCs in parallel on a single ATE. Therefore, efficient utilization of tester channels is necessary to control rapidly escalating test costs.

Research on optimization of dedicated test-access mechanisms (TAMs) has demonstrated that pin-count-aware test schedule optimization can greatly reduce test time for a given total number of pins [5], [6]. However, prior work on test scheduling for NoCs invariably assumes that the ATE interfaces with all access points using the full NoC flit width (usually 32 bits) as shown in Figure 1(a). This unnecessary restriction results in test schedules with inefficient tester channel utilization.

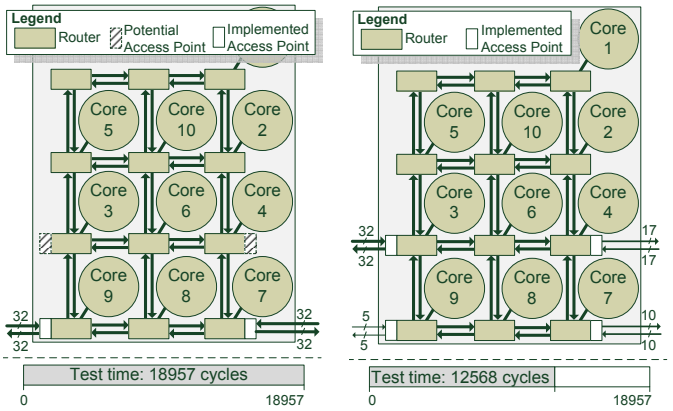
We propose to distribute a given total number of test pins (tester channels) to a set of access points, where an access point typically is assigned a smaller number of pins than the NoC's flit width. The assignment of pins to access points is done to derive a test schedule that minimizes the overall test time. The overall approach is highlighted in Figure 1(b).

Compared with TAM-based scheduling, pin-count aware NoC scheduling introduces additional constraints, e.g. link contention and access-point-to-core dependent penalties for delivery delay. This requires either to compute time-specific schedules (as opposed to assignment only for TAM) or, at least, to take delivery delay and assignment interdependence into account. Hence, while classical TAM optimization approaches can be leveraged, they cannot be directly used. Known heuristics for NoC-scheduling work on fixed flit-widths only; given the typically huge number of possible distributions of pins to access points, their direct application is also infeasible.

We propose to use MISR-based output compaction to handle test responses for embedded cores. This allows to use the same tester channels for streaming test input data to the SOC and for receiving the (short) MISR signatures, thus reducing the number of required channels by a factor of two with negligible impact on test time. Alternatively, our approach can be used in conjunction with any scan-compatible core- or access-level

*This research was supported in part by the National Science Foundation under grant no. CCF-0903392, and by the Semiconductor Research Corporation under contract no. 1992. M. Richter worked on this project while at Duke University; he is now with Intel Mobile Communications, Germany.

978-3-9810801-8-6/DATE12/©2012 EDAA



(a) Full-width access points (b) Pin-count-aware pin distribution

Fig. 1. Configuration and test time given 4 potential access points for d695 (128 tester channels)

test compression, as this kind of compression is transparent to the scheduler. The particular advantage of signature-based compaction is that it additionally reduces link contention.

The major contributions of this paper are:

1) an optimal co-optimization method for generating pin-assignments to access points and test schedules that guarantee minimum test times;
2) a heuristic solution to the co-optimization problem that can be rapidly computed;
3) the use of output compaction to reduce either pin count (for a fixed test time) or test time (for fixed pin counts).

We show that all four approaches – optimal co-optimization and heuristic co-optimization with and without output compaction – significantly outperform the baseline approach of assigning multiples of the NoC's flit width to access points.

The rest of the paper is organized as follows. Section II presents an overview of NoC basics and related prior work. Section III describes details of the proposed test schedule optimization technique and problem formulation. Section IV highlights solution techniques based on integer linear programming. Section V presents experimental results, and finally, Section VI concludes the paper.

II. Background

A network-on-chip consist of three basic components: routers, links, and network adapters [4]. *Routers* are nodes that route data according to a routing protocol. They are connected through *links* made up by one or more logical or physical channels. *Network adapters* decouple cores and routers by translating between the core's and router's communication protocol. Data streams are broken down into *packets*, which contain information about their destination. Packets in turn are made up of *flits* (flow-control units), which are the atomic units transported between routers. The *network topology* describes the logical layout of the NoC, i.e., it defines the links between routers and the routers that cores are connected to. The path a packet takes from sender to receiver is determined by the *routing algorithm*. The most popular forwarding strategy is *wormhole switching*, in which paths are set up by the header flit and torn down by the tail flit of the packet routed. While a path is active, no other packet can use links on that path (*link contention*).This limitation can be mitigated if the NoC offers *virtual channels* (VCs), i.e. provides several separate logic channels over a shared physical channel. Maintaining VCs requires independent buffer queues for each logic channel; the associated buffers make VCs an expensive feature that most current NoC do not offer [7].

Testing of the NoC infrastructure and its reconfiguration in the presence of faults have been studied recently [8], [9]. Here we focus on the use of the NoC as the test-access mechanism. A major challenge for efficient testing is to devise scheduling algorithms that optimally use the given NoC for test data transportation. Since test scheduling aims to increase test parallelism in the presence of resource conflicts, it is an $\mathcal{NP}$-hard problem [10].

Scheduling algorithms proposed in the literature can be classified into *packet-centric* and *core-centric* approaches. The first packet-centric algorithm was presented in [11] and subsequently extended to cope with power constraints [12], [13]. Core-centric approaches can also handle precedence constraints and shared BIST resources [10].

If the NoC can be driven faster than the test clock and supports virtual channels, its bandwidth can be divided into several distinct time slots to increase test parallelism [14]. To reduce power consumption, power-hungry cores can be clocked at lower speeds, while small cores may be run at faster clocks to improve flit utilization. The different core clock speeds can also be used for time-sharing common links between different cores [15]. Co-optimization of the test of the NoC and the test of cores was presented in [16]. A hybrid core-packet scheduling method encompassing power and precedence constraints, as well as support for BISTed cores and test preemption was described in [17].

A detailed test wrapper design for the reuse of functional interconnects (NoC or buses) was presented in [18], [19]. In contrast to classical TAM optimization, the number of bits in a flit – and hence the parallel data width for a given NoC – is fixed, as it is determined by performance requirements for the functional mode. Sources of the resulting flit-bit under-utilization are identified in [20].

In [21], flit utilization is increased by using varying parallel-to-parallel load ratios during pattern transfer. Another wrapper design, which achieves optimal flit utilization for all data widths at the cost of additional flip-flops and a more complex shift control logic, is outlined in [22]. It has not been shown, though, how these approaches can be leveraged to lower test time or test data volume. In [23], a scheduling algorithm is introduced that chooses, for each core, one out of a number of different possible parallel-to-parallel conversion configurations. However, it makes the restrictive assumption that the NoC supports a large number of different time slices – as many as there bits in each flit.

To reduce the number of required ATE output channels and the test input data volume, the use of input vector compression

has been proposed in [24]. In this approach, compressed test data is transferred from the ATE to the network interface, where it is decompressed before being sent over the NoC. To the best of our knowledge, that is the only work aimed at a reduced test pin count. The optimization approach described in this paper can be combined with [24] for further reducing test time. The overhead for on-chip decompression can be minimized by re-using the CRC hardware on an NoC [25].

In [26], the NoC is divided it into several partitions that each contain a single access point acting as test source and sink. Partitions are created such that each access point can communicate with the routers in its partition without interfering with other access points' communication with their respective cores. The problem of optimal partitioning with an optimal ATE pin distribution, however, has not been addressed.

III. Pin-Count-Aware Test Schedule Optimization

Today's typical NoC is based on a grid topology, implements dimension-order routing, and uses wormhole switching without virtual channels [7]. We assume an NoC with *deterministic* wormhole routing and prevent link contention by assuming that links cannot be shared between packets. Links can only be shared if the NoC supports virtual channels (the vast majority do not) and test data is sent in bursts of flits (which would require buffering at the cores and the ATE interface). We do not impose any further restrictions on the routing algorithm or the network topology.

In this paper, we do not explicitly address the design of the DfT components. We assume a DfT implementation in which test output data is sent back to the access point that sends the test stimuli (as in [19]). Furthermore, we create core wrappers using the *design_wrapper* algorithm [5] and add one additional cycle each for header and tail flit generation.

Our goal is to find a partition of a total number of tester pins to a given set of access points to allow a test schedule of minimum length.

General test scheduling problem (GTS): Given an NoC-based SOC having $|C|$ cores, $|A|$ access points, and P available test pins, determine a distribution of the P pins to the $|A|$ access points, and a *start time and access point* for the test of each core such that no link contention occurs and the overall test time is minimized. □

The decision version of GTS has been shown to be equivalent to be $\mathcal{NP}$-complete resource-constrained multiprocessor scheduling problem [10]. The feasibility of a schedule has a tight dependence on the time frames during which cores are tested via different access points to prevent link contention.

To address SOCs with a higher number of cores, we also investigated a more restricted problem. Consider the set L_1 of all links used for communication between an access point a_1 and any of its assigned cores. Then only links in L_1 will be reserved at any time during test for testing cores assigned to a_1. Thus, no link contention can occur as long as the sets of links $L_1, L_2, \ldots$ required for testing cores assigned to access points $a_1, a_2, \ldots$, respectively, are pairwise non-overlapping. By restricting the construction of assignments to those that ensure pairwise non-overlapping link sets, there is no need to rely on the computation of exact start times to prevent link contention. This restricted problem can be formulated as follows.

Contention-free assignment problem (CFA): Given an NoC-based SOC having $|C|$ cores, $|A|$ access points and P available test pins, determine a distribution of the P pins to the $|A|$ access points, an assignment of the cores to the access points such that the sets of links used for communication between the access points and their assigned cores are pairwise non-overlapping, and that the overall test time (under this restriction) is minimized. □

Since CFA limits the search space, it might lead to higher test times than GTS. The decision version of CFA is also $\mathcal{NP}$-complete, since the validity of any solution can be checked in polynomial time, and the $\mathcal{NP}$-complete *multiprocessor scheduling problem* [27] can be reduced to it.

We present two different scheduling methods, called *optimal scheduling* and *contention-free assignment*, that optimally solve GTS and CFA, respectively. Any solution to CFA can be easily converted into a test schedule by testing all cores assigned to an access point in an arbitrary order. Since optimal solutions to CFA are not guaranteed to be optimal solutions for GTS, *contention-free assignment* is a heuristic for solving GTS.

IV. ILP Problem Formulations

By expressing our optimization problems as integer linear programming (ILP) models, we can leverage ILP solvers to solve them optimally.

A. Common data structures and variables

As basic input data, we have the set of access points A, the set of cores C, the set of pin widths W available for access points, and the number P of test (input) pins available. The test length table TL encodes the number of cycles required to shift test data in and out of the test wrapper of each core for a given number of pins; the signature length table SL contains number of cycles required to shift out the signature for a given number of pins. In both cases, two additional cycles are added to model header and tail flits.

Furthermore, a number of pre-computed tables encode the properties of the NoC relevant for test schedule optimization. The entries of the two-dimensional table PD (path delay) correspond to the number of cycles required to set up a path from the access point to the core (or vice versa). In line with previous publications [11], we assume a delay of three cycles per router (including the routers at both ends of the path).

We use a matrix x of binary variables (indicator variables) to keep track of the assignment of cores to access points. A core c is assigned to access point a if and only if $x_{c,a} = 1$. Each core is assigned to exactly one access point,

$$\forall c \in C : \sum_{a \in A} x_{c,a} = 1.$$

Similarly, a table p of binary variables represents the number of (input) pins w assigned to each access point a. Each access

point must be assigned a number of pins and the sum of pins assigned may not exceed the number of available pins P,

$$\sum_{a\in A, w\in W} w \cdot p_{a,w} \leq P; \;\; \forall a \in A : \sum_{w\in W} p_{a,w} = 1.$$

The variable $ctap$ (*core time on access point*) captures the time required to test core c via access point a. Its definition depends on whether MISR compaction is used (1a) or not used (1b):

$$ctap_{c,a} = \sum_{w\in W} p_{a,w} \cdot (TL_{c,w} + SL_w) + 2PD_{c,a} \quad (1a)$$

$$ctap_{c,a} = \sum_{w\in W} p_{a,w} TL_{c,w} + 2PD_{c,a}. \quad (1b)$$

The total test time T is *at least* the maximum time required for any access point to test all cores assigned to it, given the number of pins assigned to it. We express this relationship as

$$\forall a \in A, c \in C : ta_{c,a} \geq ctap_{c,a} - (1 - x_{c,a}) \cdot M, \quad (2)$$

$$\forall a \in A : T \geq \sum_{c\in C} ta_{c,a}, \quad (3)$$

where (2) is a set of helper inequalities. The term $(1-x_{c,a})\cdot M$, with M as a value much larger than the expected maximum test time, forces $ta_{c,a}$ to zero if core c is not assigned to access point a, and to the value $ctap$ otherwise.

B. Solving the Contention-free Assignment Problem (CFA)

Link contention is captured by a table CT, which contains a '1' at position c, a, d, b if any packet between a and c shares links with any packet traveling between b and d (in any direction). The constraints

$$\forall c, d \in C, c \neq d; a, b \in A, a \neq b \text{ with } CT_{c,a,d,b} = 1 : \\ x(c, a) + x(d, b) \leq 1$$

ensure that, if testing core c through access point a and core d through access point b causes link contention, at most one of these assignments is chosen.

As all assignments that lead to link contention are forbidden by the above constraints, the test time equals the maximum time needed to test all cores assigned to one access point. The inequalities (3) therefore define a lower bound on the test time.

C. Solving the General Test Scheduling Problem (GTS)

Optimal scheduling prevents link contention by assigning a start time to every core such that link contention is prevented. To that end, more fine-grained information about the timing of link contention is necessary. This information is provided by three four-dimensional tables. The tables contain timing information for shared links when sending test data to the cores (table TT), sending test signature data back from the cores (SS), or sending test stimuli to one core while the other is sending test signature data back (TS). Each entry at position c, a, d, b corresponds to a pair (t_1, t_2), encoding the number of cycles that elapse until a packet header reaches the first contended link for the first and second core/access point pair, respectively. Furthermore, decision variables $st_{c,a}$ encode the start time of data transfer from the access point a to core c.

The goal of optimal scheduling is to minimize T subject to

$$T \geq st_{c,a} + ctap_{c,a} - (1 - x_{c,a})M, \quad \forall a \in A, c \in C,$$

where M is a constant larger than the highest expected test time, and $ctap_{c,a}$ a variable denoting the time required to test c via access point a (cf. IV-A). The expression $(1 - x_{c,a})M$ forces the right hand side of the inequality to be below zero if core c is not tested via access point a.

Note that the inequalities (3) still hold, but they are no longer sufficient, as there might be forced idle times on access points to prevent link contention. However, they are decisive in helping the ILP solver to identify non-improving assignments early.

We also define variables $tl_{c,a}$,

$$tl_{c,a} = \sum_{w\in W} p_{a,w} TL_{c,w}, \forall a \in A,\, c \in C,$$

which denote the time span required to test core c via access point a depending on the number of pins assigned to a.

If there is a possible link contention between assignments of cores c and d to a and b, respectively, and c and d are indeed assigned to a and b, data transfers must be timed such that packets from/to both cores do not overlap. Hence we need to add constraints on the start times for assignments that can lead to link contention. These constraints will only be activated if the particular assignment is actually chosen. The later is true if and only if $x_{c,a} = x_{d,b} = 1$.

Non-overlapping constraints can generally be expressed as

$$X_1 \geq Z_2 \vee Z_1 \geq X_2,$$

were X_1 (Z_1) and X_2 (Z_2) are placeholders for terms that express the first cycle that a packet from c (d) arrives at the contended link and the last packet has cleared the contended link, respectively. The compound constraint

$$x_{c,a} + x_{d,b} = 2 \rightarrow (X_1 \geq Z_2) \vee (Z_1 \geq X_2),$$

can be concisely expressed with the two linear inequalities

$$X_1 + (3 - x_{c,a} - x_{d,b} - o)M \geq Z_2, \text{ and}$$
$$Z_1 + (2 - x_{c,a} - x_{d,b} + o)M \geq X_2,$$

where o is a binary variable only used for these two inequalities, and M is a constant value guaranteed to be greater than all possible values of Z_2 and X_2.

The formulation of X_1, X_2, Z_1 and Z_2 depends on the considered type of conflict (between test stimuli, test signatures, or test stimuli and test signatures). For brevity, we only present the formulation for conflicts between test stimuli and test signatures. Considering the case without MISR, we define our placeholders X_1, X_2, Z_1 and Z_2 as

$$X_1 := st_{c,a} + TS_{c,a,d,b}[1],$$
$$X_2 := st_{c,a} + tl_{c,a} + TS_{c,a,d,b}[1],$$
$$Z_1 := st_{d,b} + PD_{d,b} + TS_{c,a,d,b}[2],$$
$$Z_2 := st_{d,b} + tl_{d,b} + PD_{d,b} + TS_{c,a,d,b}[2].$$

If a MISR is used, the timing for the packets transporting the test signature changes, as the test signature is sent only after all test data has been shifted out. Consequently, Z_1 and Z_2 have to be adjusted:

$$Z_1 := st_{d,b} + tl_{d,b} + PD_{d,b} + TS_{c,a,d,b}[2],$$
$$Z_2 := st_{d,b} + tl_{d,b} + PD_{d,b} + TS_{c,a,d,b}[2] + SL_b.$$

Formulations for the other two conflicts types (between two test input stimuli, and two test signatures) can be derived in a similar way.

V. Experimental Results

For our experiments, we used five SOCs from the ITC'02 SOC Test Benchmarks [28]. As the benchmarks were published with dedicated TAM optimization in mind, no layout information for NoCs is provided. Assuming a grid topology, we therefore placed the cores in the grid in a left-to-right, top-to-bottom fashion. In line with previously reported results, we assumed XY-routing, a switching delay of three clock cycles per router and an additional cycle for header and tail flits, respectively. We assumed a flit width of 32 bits. We marked four routers as possible access points – two to the left and two to the right of the grid.

We run two types of experiments to address the following two DfT scenarios: a) given an upper bound on the total number of pins, the test time is to be minimized; b) given an upper bound for the test time, the total number of required pins is to be minimized. Finally, we computed optimal schedules for different core placements for d696 and g1023 (10 and 14 cores, respectively) and compared them to schedules generated for CFA.

A. Fixed Total Number of Pins

To evaluate pin-count-aware scheduling with and without MISR compaction, we compare the resulting test time for the traditional full-width approach, pin-count-aware scheduling, and MISR-based pin-count-aware scheduling for 32+32 and 64+64 pins (input+output). In the full-width approach, this implies choosing the single best or the best combination of two access points, requiring 64 and 128 pins, respectively. The schedules for full-width access points that we use as the baseline are provably optimal. The pin-count-aware schedules were allowed to use all four access points, as long as sum of pin counts did not exceed the maximum number of pins. Pin-count-aware scheduling is based on the contention-free assignment heuristic.

Table I displays the duration of the shortest possible test schedule for each of the three approaches given 64 ATE channels, i.e. one full access point for the traditional, full-width approach. In the second column ("Full-Width"), the number of clock cycles of the shortest schedule for the full-width (baseline) approach are given. Under the third and fourth major headings, the absolute length ("Cycles") as well as the length relative to the full-width approach ("Rel.") are given for the pin-count-aware and the pin-count-aware schedules with MISR, respectively. For all considered benchmarks, the pin-count-aware scheduling methods significantly reduce test time (by 25%–65% for all but t512505). Employing output compaction offers a substantial improvement both compared to the full-width and the pin-count-aware scheduling without compaction, bringing test times down by 50%-70% compared to the full-width approach. For t512505, a single large bottleneck core prevents significant improvements for pin-count-aware scheduling without compaction. With compaction, more pins can be used for data input, which allows to dedicate an access point with 28 pins to the bottleneck core, thus reducing test time by 50%.

TABLE I
Test Cycles with 64 ATE channels

SOC	Full-Width	Pin-Aware		P-A + MISR	
	Cycles	Cycles	Rel. (%)	Cycles	Rel. (%)
d695	37869	22195	58.61	12598	33.27
g1023	50397	17947	35.61	14815	29.40
p22810	543449	240178	44.20	151273	27.84
t512505	11100481	10446742	94.11	5228440	47.10
p93791	1224469	912781	74.55	467561	38.18

TABLE II
Test Cycles with 128 ATE channels

SOC	Full-Width	Pin-Aware		P-A + MISR	
	Cycles	Cycles	Rel. (%)	Cycles	Rel. (%)
d695	18957	12568	66.30	10115	53.36
g1023	25181	14808	58.81	14814	58.83
p22810	271654	151203	55.66	139813	51.47
t512505	5550278	5228434	94.20	55228440	94.20
p93791	612143	467441	76.36	318531	52.04

The results for 128 test pins are shown in Table II. The relative performance of the three scheduling methods is comparable to the observations made for 64 pins. The CPU times for the test cases in Tables I-II using the FICO™Xpress solver ranged from 0.3 to 247 seconds on a computer with a 3.0 GHz Intel Xeon CPU with 9GB memory.

B. Minimum Pin Count

In the second set of experiments, we determined the minimum number of pins required to achieve a test time *lower than or equal to* full-width scheduling. Table III lists the minimum number of pins for pin-count-aware scheduling without ("Pin-Aware") and with output compaction ("P-A + MISR") to match the test times corresponding to 64 pins with full-width scheduling. Both pin-count-aware approaches allow a significant reduction of the number of pins. The required number of pins for pin-count-aware scheduling without compaction is in the range of 30%–75% of full-width scheduling; with compaction the number of required pins falls to 20%–40% of the full-width approach.

C. Optimal Schedules

To experimentally assess the quality of the heuristic method, we created and compared both optimal schedules and schedules based on contention-free assignment. For the comparison,

TABLE III
PIN COUNT TO MATCH 64 CHANNEL FULL-WIDTH TEST TIMES

SOC	Pin-Aware		P-A + MISR	
	Pins	Rel. (%)	Pins	Rel. (%)
d695	38	59.38	19	29.69
g1023	20	31.25	11	17.19
p22810	26	40.63	13	20.31
t512505	40	62.50	20	31.25
p93791	48	75.00	24	37.50

TABLE IV
TEST LENGTH OVERHEAD OF PROPOSED HEURISTIC (IN %)

	d695			g1023		
Channels	Avg.	Median	Max.	Avg.	Median	Max.
64	1.48	0.83	5.57	6.28	6.29	14.35
64 +MISR[1]	12.12	13.37	24.77	0.90	0.01	12.22
128[2]	8.81	9.79	22.12	0.03	0.00	1.83
128 + MISR	3.15	2.35	12.25	0.87	0.00	12.11

[1]For *g1023*, 98 out of 100 runs finished within the limit of 180 min
[2]For *g1023*, 90 out of 100 runs finished within the limit of 180 min.

we used *d695* and *g1023* (11 and 14 cores, respectively). We created 100 different NoCs for each of the two benchmarks by randomly permuting the order in which cores are assigned to routers. We then created schedules for 64 and 128 channels without and with output compaction by MISR, i.e., for the scenarios presented in Section V-A. The relative differences of the test lengths for optimal and heuristic scheduling are displayed in Table IV.

For both benchmarks, we show the average and the median of the relative differences as well as the maximum difference encountered. For example benchmark *g1023* with 64 ATE channels and no output compaction, the contention-free assignment-based test schedules are on average 6.28% longer than the optimal schedule.

CPU times for solving the CFA did not exceed 2.5 sec for any configuration of the two SOCs. In contrast, the maximum CPU time for optimal scheduling was 32 seconds for *d695* and 292 seconds for *g1023*. Moreover, 12 of the 400 instances of *g1023* exceeded our run-time limit of 180 minutes.

VI. CONCLUSION

We have investigated the problem of minimizing test application time for NoC-based test delivery. In particular, we have proposed the use of test output compaction using MISR and optimizing the test channel distribution to access points. We have developed an ILP model that can be used to create optimal solutions. To heuristically solve this problem for large SOC instances, we introduced the more restricted contention-free assignment problem. Experimental results for the largest SOCs from the ITC 2002 Test Benchmarks demonstrate significant advantages of pin-count aware scheduling and test output compaction.

REFERENCES

[1] International Technology Roadmap for Semiconductors (ITRS) 2009. [Online]. Available: www.itrs.net/Links/2009ITRS/Home2009.htm
[2] L. Benini *et al.*, "Networks on Chips: A New SoC Paradigm," *Computer*, vol. 35, pp. 70–78, 2002.
[3] P. P. Pande *et al.*, "Performance Evaluation and Design Trade-Offs for Network-on-Chip Interconnect Architectures," *IEEE Transactions on Computers*, vol. 54, pp. 1025–1040, 2005.
[4] T. Bjerregaard *et al.*, "A Survey of Research and Practices of Network-on-Chip," *ACM Comput. Surv.*, vol. 38, June 2006.
[5] V. Iyengar *et al.*, "Test Wrapper and Test Access Mechanism Co-Optimization for System-on-Chip," *Journal of Electronic Testing*, vol. 18, pp. 213–230, 2002.
[6] S. K. Goel *et al.*, "Effective and Efficient Test Architecture Design for SOCs," *Proc. Int. Test Conf*, p. 529, 2002.
[7] E. Salminen *et al.* (2008, March) Survey of Network-on-Chip Proposals. Tech. Rep. [Online]. Available: www.ocpip.org/socket/whitepapers
[8] P. P. Pande *et al.*, "Design, Synthesis, and Test of Networks on Chips," *IEEE Design & Test of Computers*, vol. 22, no. 5, pp. 404–413, 2005.
[9] A. Dalirsani *et al.*, "Structural Test for Graceful Degradation of NoC Switches," in *Proc. IEEE European Test Symp*, 2011, pp. 183–188.
[10] C. Liu *et al.*, "Test Scheduling for Network-on-Chip with BIST and Precedence Constraints," in *Proc. Int. Test Conf.*, 2004, pp. 1369–1378.
[11] E. Cota *et al.*, "The Impact of NoC Reuse on the Testing of Core-Based Systems," in *Proc. VLSI Test Symp*, 2003, pp. 128–133.
[12] ——, "Power-Aware NoC Reuse on the Testing of Core-Based Systems," in *Proc. Int. Test Conf*, 2003, pp. 612–621.
[13] ——, "Reusing an On-Chip Network for the Test of Core-Based Systems," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 9, pp. 471–499, Oct. 2004.
[14] J. M. Nolen *et al.*, "Time-Division-Multiplexed Test Delivery for NoC Systems," *IEEE Design & Test of Computers*, vol. 25, no. 1, pp. 44–51, 2008.
[15] C. Liu *et al.*, "Power-Aware Test Scheduling in Network-on-Chip Using Variable-Rate On-Chip Clocking," in *Proc. IEEE VLSI Test Symp*, 2005, pp. 349–354.
[16] ——, "Reuse-Based Test Access and Integrated Test Scheduling for Network-on-Chip," in *Proc. Design, Automation and Test in Europe*, 2006.
[17] E. Cota *et al.*, "Constraint-Driven Test Scheduling for NoC-Based Systems," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 11, pp. 2465–2478, 2006.
[18] A. M. Amory *et al.*, "Wrapper Design for the Reuse of Networks-on-Chip as Test Access Mechanism," in *Proc. IEEE European Test Symp*, 2006, pp. 213–218.
[19] ——, "Wrapper Design for the Reuse of a Bus, Network-on-Chip, or Other Functional Interconnect as Test Access Mechanism," *IET Computers & Digital Techniques*, vol. 1, no. 3, pp. 197–206, 2007.
[20] A. van den Berg *et al.*, "Bandwidth Analysis for Reusing Functional Interconnect as Test Access Mechanism," in *Proc. IEEE European Test Symp*, 2008, pp. 21–26.
[21] M. Li *et al.*, "An Efficient Wrapper Scan Chain Configuration Method for Network-on-Chip Testing," in *Proc. IEEE Computer Society Annual Symp. Emerging VLSI Technologies and Architectures*, 2006.
[22] F. A. Hussin *et al.*, "Optimization of NoC Wrapper Design Under Bandwidth and Test Time Constraints," in *Proc. IEEE European Test Symp.*, 2007, pp. 35–42.
[23] J. Li *et al.*, "Channel Width Utilization Improvement in Testing NoC-Based Systems for Test Time Reduction," in *Proc. IEEE Int. Symp. Electronic Design, Test and Applications DELTA*, 2008, pp. 26–31.
[24] J. Dalmasso *et al.*, "Improving the Test of NoC-Based SoCs with Help of Compression Schemes," in *Proc. IEEE Computer Society Annual Symp. VLSI*, 2008, pp. 139–144.
[25] V. Froese *et al.*, "Reusing NoC-Infrastructure for Test Data Compression," in *Proc. VLSI Test Symp.*, 2010, pp. 227–231.
[26] A. M. Amory *et al.*, "DfT for the Reuse of Networks-on-Chip as Test Access Mechanism," in *Proc. IEEE VLSI Test Symp*, 2007, pp. 435–440.
[27] G. Ausiello *et al.*, *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer, 1999.
[28] E. J. Marinissen *et al.*, "A Set of Benchmarks for Modular Testing of SOCs," in *Proc. Int. Test Conf*, 2002, pp. 519–528.