A Proposal for Routing-Based Timing-Driven Scan Chain Ordering*

Puneet Gupta*, Andrew B. Kahng*‡ and Stefanus Mantik †
* Department of Electrical and Computer Engineering, UC San Diego, La Jolla, CA, USA
‡ Department of Computer Science and Engineering, UC San Diego, La Jolla, CA, USA
† Cadence Design Systems, Inc., San Jose, CA, USA
{puneet@ucsd.edu, abk@ucsd.edu, smantik@cadence.com}

Abstract

Scan chain insertion can have large impact on routability, wirelength and timing. We propose a routing-driven and timing-aware methodology for scan insertion with minimum wirelength. We take into account timing slacks at all sinks that are affected by scan insertion, to achieve a scan chain ordering that meets timing and has smallest wirelength. For the case where sink timing is not met, we also propose a buffer insertion methodology with minimum wirelength objective. The key contribution of this paper is a method to compute a timing-driven incremental connection suited to scan insertion; this has possible applications in general incremental routing.

1 Introduction and Motivation

In VLSI design for testability, a *scan chain* is commonly used to implement the shift registers that store the input and output vectors during the testing phase of manufacturing. Registers in the scan chain are connected as a single path, with ends of the path connected to *primary input* (PI) and *primary output* (PO) pads. Test input values are shifted into the registers through the PI pad; then, a test is performed and the test output values are shifted out through the PO pad. Figure 1 illustrates a scan chain.

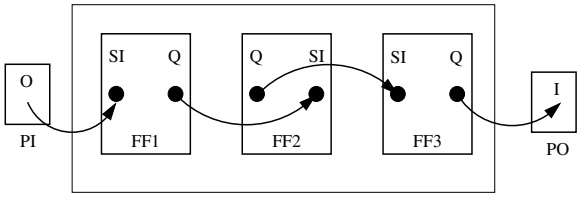

Figure 1: Example of a scan chain with three scan registers $c1$, $c2$, and $c3$. "SI" and "Q" are used to distinguish between the scan-in and scan-out pins in each sequential cell.

A primary objective of design for testability is to minimize the impact of test circuitry on chip performance and cost. Thus, it is essential to ensure that the design remains timing feasible after scan insertion. At the same time, small wirelength overhead of scan chains is also desirable: this increases wirability and/or reduces chip area while increasing signal speed by reducing capacitive loading effects on nets that share register pins with the scan chain.

Several previous works have performed scan chain reordering based on layout (placement) information. The scan chain ordering problem is transformed to a symmetric or asymmetric traveling salesman problem (TSP). Feuer and Koo [5] wrote perhaps the first published work showing how TSP heuristics can be applied to scan chain optimization. Previous placement-based scan chain ordering approaches compute the cost of stitching one flip-flop to another as either cell-to-cell Manhattan distance [7], [11], [1] or pin-to-pin Manhattan distance [2], [9]. The former metric gives a symmetric TSP while the latter gives rise to an almost symmetric TSP [2]. The fundamental assumption in most current work on layout-driven scan chain ordering is that the wirelength overhead due to scan insertion is equal to the Manhattan distance between the scan-in and scan-out pins of the flip-flops. However, this assumption is incorrect: the scan connection need only reach the output *net*, not the output *pin*. [10] orders the scan chain after global routing but uses channel congestion as its objective, making it applicable only in the channel routing context.

[12] proposes a routing-based flow for scan chain ordering that uses the *incremental* routing cost (connecting to existing routing, rather than to the output pin) as the cost measure for a scan connection. This is in contrast to existing placement-based methods which simply use the Manhattan distance from the flip-flop output pin to scan-in pin of the other flip-flop as the cost measure. Under their formulation, the resulting asymmetric traveling salesman problem (ATSP) may be highly non-metric. In Figure 2, the existing route is shown by solid lines while potential scan connection routes are shown by dotted lines. We label the possible scan connection routes by their respective lengths, $1, 2, 3$ and 4. We note the following.

- The cost of connecting the Q output of FF A to the SI pin of FF B (denote this by AB) is given by the length of the routing segment 1, which is much less than the total Manhattan distance between the corresponding pins. Thus, a placement-based approach will inaccurately estimate the cost of making this scan connection.
- This formulation of the TSP can be highly asymmetric. For example, $BA(=3) \gg AB(=1)$.
- We can even get non-metric TSP instances (i.e., the triangle inequality may not hold). For example, $AB(=1)+BC(=2) < AC(4)$.

*This work is supported in part by the MARCO Gigascale Research Center and by Cadence Design Systems, Inc.

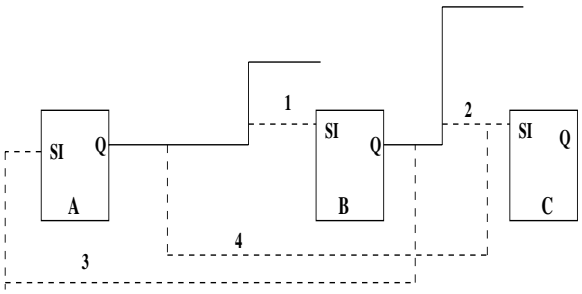

Figure 2: An example showing the highly asymmetric and non-metric nature of the ATSP when doing incremental scan insertion.

[12] shows very good improvements in wirelength compared to the placement based approach. However, the approach of [12] is timing oblivious. Using the unused output pin of the scan flip-flop to make the scan connection is one of the ways in which industrial flows try to keep the timing impact of scan insertion under control. We have noticed that up to 60% of the scan nets fall into this category in some industrial benchmark designs. This puts unnecessary constraints on design and synthesis besides increasing the total wirelength overhead of the scan insertion. Since smaller wirelength does not necessarily imply better timing, we extend the approach of [12] to account for timing slacks on all the sink pins. We also consider inclusion of buffers in the scan connection to meet the timing constraints. We resolve possible clashes in use of available buffer sites by solving an assignment problem.

In the remainder of this paper, we discuss our incremental connection technique with application to scan insertion. Section 2 describes the core of our approach namely, (i) a method to compute timing-driven incremental connection and (ii) a buffer insertion method when timing is not met. Experiments are described in Section 3 with conclusions given in Section 4.

2 Timing-Driven Scan Chain Ordering

In this section, we propose a routing-based timing-driven approach to scan chain ordering with minimum wirelength objective. The timing-awareness entails taking into account timing slacks at all relevant sinks. We first present the core of such a method, namely, finding the minimum wirelength incremental connection which meets all timing requirements. We then describe a buffer insertion technique for connections which do not meet timing requirements.

2.1 Computing optimal attachment point

Adding scan connections can cause additional timing violations. We give a method to compute the route with least wirelength to attach the *SI* pin of ff_{in} to the output of ff_{out}, such that no timing constraints are violated on any of the sinks of the fanout tree of ff_{out}. We first divide the fanout routing tree of the FF output into *optimization segments*. Optimization segments are the segments between any two consecutive Steiner points/source/sink/bends. Each optimization segment is either entirely horizontal or entirely vertical. The *beginning* of an optimization segment is defined as the point of the segment topologically closest to the source, while the segment *end* is topologically closest to the sink. The *influenced sinks* of a segment are the sinks that use the segment as a part of their (unique) route to the source.

We seek the best attachment point on the optimization segment, such that the route from the *SI* pin of ff_{in} to the optimization segment has minimum wirelength among all possibilities which do not violate sink timing requirements. We use the Elmore delay approximation; since it is an upper-bound on 50% threshold delay for RC trees [6], the solution is guaranteed to meet timing though we may overestimate the timing overhead of scan insertion. Define the following :

- t_i = arrival time slack at sink i for the FF output under consideration.
- *root* = root of the routing tree, i.e., Q or $\bar{Q}$ pin of ff_{out}.
- o_j = optimization segment j under consideration.
- $l(M,N)$ = routing distance from point M to point N.
- $begin_j$ = location of the point on o_j which is topologically closest to *root*.
- end_j = location of the point on o_j which is topologically farthest from *root*.
- b_{ij} = location of the point on route from *root* to i which is topologically closest to o_j.
- s_j = Steiner point added to connect *SI* to the output pin of ff_{out}.
- v_j = point at which minimum-wirelength route from *SI* pin to o_j intersects o_j.
- x_j = routing distance (entirely horizontal or entirely vertical routing only) of s_j from v_j. It locates the attachment point.
- r = wire resistance per unit length.
- c = wire capacitance per unit length.
- C_L^{SI} = input capacitance of the *SI* pin.
- $n_Q(n_{\bar{Q}})$ = number of optimization segments in the fanout tree of $Q(\bar{Q})$. This is equal to the number of sink pins and the number of Steiner points and bends in the fanout routing tree of $Q(\bar{Q})$.
- $dist_Q(dist_{\bar{Q}})$ = length of the shortest timing-feasible route for adding the *SI* pin on the fanout tree of $Q(\bar{Q})$ output of ff_{out}.
- $n_Q^s(n_{\bar{Q}}^s)$ = number of sinks in the fanout tree of $Q(\bar{Q})$.

Figure 3 illlustrates above definitions. Consider incremental insertion of the *SI* pin into the fanout routing tree of the Q pin of ff_{out}. QC, C2, CD, D1 are the four optimization segments. The optimization segment under consideration is D1 and sink 1 is its only influenced sink.

The delay increase on an influenced sink p of an optimization segment o_j due to addition of a $Q-SI$ route

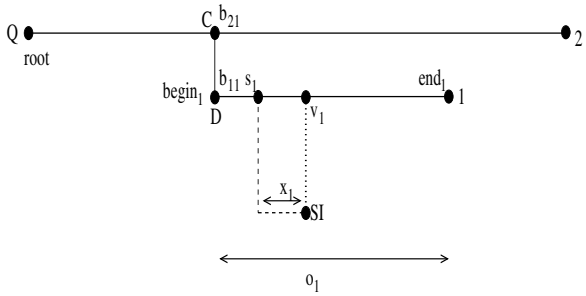

Figure 3: Incremental scan-in insertion

(connecting to o_j at point s_j) is given by

$$d_{p,j} = r(l(root,b_{pj}) + l(begin_j,v_j) - x_j) * (c(x_j + l(SI,v_j)) + C_L^{SI}) \quad (1)$$

where $d_{p,j}$ denotes the increase in Elmore delay from Q to sink p. For there to be no timing violation we require $d_{p,j} \leq t_p$ for all influenced sinks p, i.e.,

$$r(l(root,b_{pj}) + l(begin_j,v_j) - x_j) * (c(x_j + l(SI,v_j)) + C_L^{SI}) \leq t_p \quad (2)$$

In case there exist more than one influenced sinks, we use t_p the smallest among the slacks. Similarly, delay on any other sink q (i.e., any uninfluenced sink) must satisfy

$$d_{q,j} = r(l(root,b_{qj})) * (c(x_j + l(SI,v_j)) + C_L^{SI}) \leq t_q \quad (3)$$

$$0 \leq x_j \leq l(begin_j,v_j) \quad (4)$$

Note that negative x_j is not an admissible solution because for any $x_j < 0$, $-x_j$ always gives a smaller increase in delay for the influenced sink while the delay at uninfluenced sinks remains unchanged. Equations (2), (3) and (4) give at most $n_Q^s + 1$ constraints; the objective is to find minimum $(x_j + l(SI,v_j))$ which obeys these constraints. From Equation (3) we get for optimization segment o_j that

$$x_j \leq x_{q_{max},j} = (\frac{t_q}{rcl(root,b_{qj})} - \frac{C_L^{SI}}{c} - l(SI,v_j)) \quad (5)$$

where q is any sink not influenced by o_j. $x_{q_{max},j}$ represents the maximum value x_j can have without violating timing constraints on sink q. If $x_{q_{max},j} < 0$ for any q not influenced by o_j then the SI pin cannot be attached to o_j without a buffer. From Equation (2) we get

$$x_j^2 + x_j k_0 - k_1 \geq 0 \quad (6)$$

where $k_0 = (l(SI,v_j) + C_L^{SI}/c - l(root,b_{pj}) - l(begin_j,v_j))$ and
$k_1 = (C_L^{SI}/c + l(SI,v_j))(l(root,b_{pj}) + l(begin_j,v_j)) - t_p/rc$.

For inequalities (6) and (4) to hold, either the determinant of the quadratic expression should be negative or at least one of the roots of the quadratic expression should lie between 0 and $l(begin_j,v_j)$. There are three cases:

$$k_0^2 + 4k_1 < 0 \quad (7)$$

or

$$0 \leq (-k_0 + \sqrt{k_0^2 + 4k_1})/2 \leq l(begin_j,v_j) \quad (8)$$

or

$$0 \leq (-k_0 - \sqrt{k_0^2 + 4k_1})/2 \leq l(begin_j,v_j) \quad (9)$$

If none of (7), (8) and (9) is satisfied, then a buffer needs to be inserted at some cost (call it M). Figure 4 summarizes the calculation of x_j.

If $x_{q_{max},j} < 0$ **for some** q **not influenced by** o_j **then**
$x_j = M$ **where** M **is a large number**
else
  if (7) or (9) is satisfied then
  $x_j = 0$
  else
    if (8) is satisfied then
    $x_j = \frac{-k_0 - \sqrt{k_0^2+4k_1}}{2}$
    else $x_j = M$.

Figure 4: Computation of x_j = position of optimal attachment point

This computation must be performed for all the optimization segments in the fanout trees of ff_{out} outputs Q and $\bar{Q}$. Then e.g., $dist_Q$ is given by

$$dist_Q = min_{1 \leq j \leq n_Q}(x_j + l(SI,v_j)) \quad (10)$$

and the cost of the corresponding ATSP edge is given by

$$dist(O,I) = min(dist_Q, dist_{\bar{Q}}) \quad (11)$$

If $dist(O,I) \geq M$, then no pure wire solution exists, and a buffer needs to be inserted.

Time Complexity

$O(n_Q^s)$ operations are required to determine the optimal x_j for any given optimization segment o_j. There are n_Q optimization segments for Q, hence $O(n_Q^s n_Q)$ time is required to determine $dist_Q$, and similarly $O(n_{\bar{Q}}^s n_{\bar{Q}})$ time is required to determine $dist_{\bar{Q}}$. Thus, $O((n_Q^s n_Q + n_{\bar{Q}}^s n_{\bar{Q}}))$ time is required to determine the cost of one edge in the ATSP instance.

2.2 Buffer Insertion

In the above ATSP solution, the edges with cost M in the solution correspond to routes where a buffer needs to be inserted. For buffer insertion, the same procedure as in the previous subsection is followed, with the SI pin replaced by the buffer location. Buffer insertion costs are determined for the k buffer locations nearest to the source FF. Given m edges that require a buffer and f buffer locations available, we need to assign a buffer location to each edge such that total wiring overhead is minimized while timing constraints are satisfied. The maximum Manhattan distance of a feasible buffer location from the source FF is upper-bounded by the minimum y obtained by varying x in Equations (2) and (3). A simple sanity check is that the buffer location cannot be farther than the

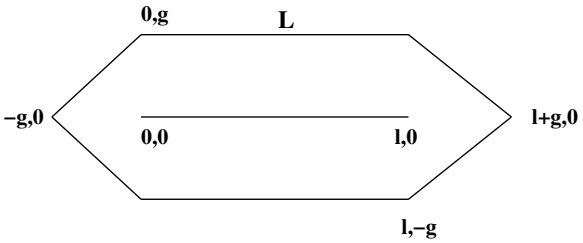

Figure 5: All the admissible buffer sites lie within the bounding box L. In the figure $g = \frac{C_L^{SI} - C_L^B + cl(SI,o_j)}{c}$

SI pin itself. In particular, if $l(B,o_j)$ (resp., $l(SI,o_j)$) is the minimum Manhattan distance of a buffer location B (resp., the scan-in pin *SI*) from the optimization segment o_j then

$$l(B,o_j) < \frac{C_L^{SI} - C_L^B + cl(SI,o_j)}{c} \quad (12)$$

where C_L^B is the input capacitance of the buffer. This locus is shown in Figure 5.

For each buffer site (we assume possible buffer sites to be given) we calculate its optimal location as in the previous section.[1] Then, the cost of assigning buffer B_k to an ATSP edge (ff_{out}, ff_{in}) is given by the total wirelength of the route (i.e., the sum of wirelengths from ff_{out} output to B_k and from B_k to the scan-in of ff_{in}).

After the assignment problem is solved, we have the complete scan chain solution along with all necessary buffers and their locations. The additional time required to calculate the assignment costs for k buffers for each of the m FF pairs (edges) is $O(k \times (n_Q^s(n_Q) + n_{\bar{Q}}^s n_{\bar{Q}}))$

3 Experiments

In this section we describe our experimental setup and the test cases. We use Cadence *Silicon Ensemble v5.3.125* (SE) and Cadence *QPlace v5.1.68* as the physical design tool to perform the industry placement-based scan chain ordering. In addition, we use Cadence *WRoute v2.2.31* for the routing tool. We have developed basic utilities for extracting the industry tool's scan ordering from a routed DEF (the order is not otherwise available in the output DEF), for generating pin-to-pin distances from the placed DEF, for generating minimum pin-to-net distances from the routed DEF, and for plugging a solver-generated scan order into DEF for routing. [12] provides a summary of ATSP solvers; they conclude that *ScanOpt* [3] is a good ATSP solver for the purpose of scan-chain ordering, and we also use *ScanOpt* in all our experiments. The basic elements of our flows are given below.

1. `Initial QPlace:` Design is placed with *QPlace* to generate a placed DEF netlist.
2. `Placement-Based Scan Order:` We extract scan flip-flop locations from the placed DEF. We compute pairwise pin-to-pin distances to construct the TSP cost matrix. The ATSP solver (ScanOpt) is then used to obtain a scan chain order.
3. `WRoute:` The placed (or partially routed) netlist is routed using *WRoute*.
4. `Routing-Based Scan Order:` As in [12] we extract fanout routing trees of all the scan flip-flops from the routed netlist. The ATSP cost matrix is computed from the minimum pin-to-tree distances. The ATSP solver then computes the routing-based scan order.
5. `Timing-Driven Scan Order:` We compute slacks at all sinks of fanout routing trees of scan flip-flops and find out the best $timing - feasible$ attachment point for the route from either the Q or $\bar{Q}$ pin of the output flip-flop to the *SI* of the input flip-flop. The ATSP cost matrix is constructed from these minimum timing-feasible pin-to-tree distances. The ATSP solver then computes the timing-aware scan order.

From these elements, we construct the following scan chain insertion flows.

- *Flow I: 1, 2, 3* ScanOpt placement based scan ordering flow.
- *Flow II: 1, 3, 4, 3* Routing-based flow as in [12].
- *Flow III: 1, 3, 5, 3* Our proposed timing-aware routing-based flow.

We have studied a single design obtained from industry sources. This test case was obtained in LEF/DEF format and then modified to merge its multiple scan chains into one scan chain. We then generated alternate placements for the test case by (1) randomly swapping some (30 to ensure routability) scan flip-flop locations and (2) increasing area of the site map by 20%. Parameters of the resulting test cases are given in Table 1; A_{swap} denotes the test case with placement of A altered by randomly swapping the placements of scan flip-flops, and A_{expand} denotes the test case obtained by increasing the site map of A.

Test Case	# Cells	# Scan FFs	Die Area mm^2	# Metal Layers
A/A$_{swap}$	6390	1226	0.526	4
A$_{expand}$	6390	1226	0.632	4

Table 1: Characteristics of the test cases.

To gain some intuition regarding the impact of timing-awareness on scan odering, we study the resulting tour structues. Define the *dissimilarity* between two scan chain orderings as the percentage of edges which differ in the two corresponding TSP tours. In other words, if two n-edge tours have $m \leq n$ edges in common, the dissimilarity between them is given by $\frac{n-m}{n} \times 100\%$. Table 2 shows that scan chain orders generated using the different cost metrics differ significantlyin structure. This suggests that the timing-oblivious wirelength minimization during scan chain ordering has less chance of achieving a timing-feasible result.

[12] reports up to 80% reduction in scan wirelength overhead when moving from a placement-based ordering (Flow I) to a routing based ordering (Flow II).[2] Since the

[1] Optimal location for each buffer is obtained by selecting the site that gives the minimum pin-to-net distance (i.e., buffer pin to scan net) from all available buffer sites. Optimization segments for which the buffer site does not fall inside the admissible locus (12) are ignored.

[2] For example, for test case A, the wirelength without scan is 864765 μm and scan overhead for the placement based flow is 21502 μm (i.e., total wirelength with scan of 864765 + 21502 = 886267 μm), while the scan overhead for the routing based flow is 6540 μm.

Test Case	Dissimilarity		
	Flows I-II	Flows I-III	Flows II-III
A	76%	77%	57%
A_{swap}	77%	71%	65%
A_{expand}	75%	75%	52%

Table 2: Dissimilarity of scan orderings for placement-driven (Flow I), routing-driven (Flow II) and timing-driven (Flow III) scan insertion flows.

scan-in pin of any scan flipflop is unlikely to have strict timing constraints, a pin-to-pin connection will always meet timing though with a larger wirelength overhead. Therefore, we expect our timing-driven routing-based flow (Flow III), which is based on pin-to-net distances, to be at least as good as the placement based flow in terms of wirelength while giving better timing results. Comparing to a routing-driven flow, we expect our flow to have a worse wirelength but better timing (better timing implies larger minimum slack and fewer timing violations).

While we believe that the above considerations support the adoption of a timing-aware routing-based scan flow, we have not been able to confirm the benefits of our proposed flow. The timing-driven approach that we propose requires the ability to route specific pin-to-tree connections. Modern back-end data models and routers offer this capability via the *virtual-pin* ("vpin" in DEF) construct. On other hand, the presence of many constraints appears to hamper traditional routing heuristics [8]. The industry router that we use does not gracefully handle situations involving many such constraints, especially in its incremental routing mode, and quality of result appears to suffer even though the routing-based timing-driven scan ordering is "better". More specifically, we have tried to enforce routes corresponding to the incremental connection using the *VPIN* construct in the DEF format, using such approaches as:

- placing vpins on the routing grid, avoiding other pins and vpins;
- placing vpins as regions as well as points;
- routing from-scratch as well as in ECO mode; and
- prerouting the scan nets that have vpins as well as routing all nets together.

Unfortunately, the commercial router does not behave well when constrained; in our experience no routing attempt was ever completed, which we find to be a very unexpected and nonintuitive outcome.

4 Conclusions

We have proposed a technique for timing-driven routing-based *incremental* connection of a pin to a routed tree. We have also proposed a buffer insertion methodology for the connections which do not meet timing. These methods form the basis of a timing-driven routing-based scan chain ordering flow, and have further applications in the incremental routing regime. We believe that our flow can be effective in timing-feasible scan insertion with low wirelength overhead but are unable to substantiate this claim due to the unavailability of a controllable router. This may serve as further motivation for new layout tools that can handle "constraint-dominated" usage contexts in future technology nodes.

References

[1] S. Barbagello, M.L. Bodoni, D. Medina, F. Corno, P. Prinetto and M.S. Reorda, "Scan-Insertion Criteria for Low Design Impact", *Proc. VLSI Test Symposium*, 1996, pp. 26-31.

[2] K.D. Boese, A.B. Kahng and R.S. Tsay, "Scan Chain Optimization: Heuristic and Optimal Solutions", Internal Report, CS Dept., University of California at Los Angeles, October 1994. Downloaded on the WWW from *http://vlsicad.ucsd.edu/GSRC/Bookshelf/Slots/ScanOpt/*.

[3] GSRC Bookshelf, "Scan Chain Optimization", *http://vlsicad.ucsd.edu/GSRC/Bookshelf/Slots/ScanOpt/.*

[4] C.S. Chen and T.T. Hwang, "Layout Driven Selection and Chaining of Partial Scan Flip-Flops", *Journal of Electronic Testing: Theory and Applications* 13(1998), pp. 19-27.

[5] M. Feuer and C. C. Koo, "Method for Rechaining Shift Register Latches Which Contain More Than One Physical Book", *IBM Technical Disclosure Bulletin* 25(9) (1983), pp. 4818-4820.

[6] R. Gupta, B. Tutuianu and L.T. Pileggi, "The Elmore Delay as a Bound for RC Trees with Generalized Input Signals", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 16(1), 1997, pp.95-104.

[7] M. Hirech, J. Beausang and X. Gu, "A New Approach to Scan Chain Reordering Using Physical Design Information", *International Test Conference*, 1998, pp. 348-355.

[8] A. B. Kahng and S. Mantik, "On Mismatches Between Incremental Optimizers and Instance Perturbations in Physical Design Tools", *Proc. IEEE/ACM Intl. Conference on Computer-Aided Design*, November 2000, pp. 17-21.

[9] S. Kobayashi, M. Edahiro and M. Kubo, "A VLSI Scan-Chain Optimization Algorithm for Multiple Scan-Paths", *IEICE Trans. Fundamentals* E82-A(11) (1999), pp. 2499-2504.

[10] K.H. Lin, C.S. Chen and T.T. Hwang, "Layout Driven Chaining of Scan Flip-flops", *IEE Proc., Comput. Digit. Tech.*, 143(6) (1996), pp. 421-425

[11] S. Makar, "A Layout Based Approach for Ordering Scan Chain Flip-Flops", *Proc. International Test Conference*, 1998, pp. 341-347.

[12] P. Gupta, S. Mantik and A.B. Kahng, "Routing Driven Scan Chain Ordering", *Proc. Asia South-Pacific Design Automation Conference*, 2003.