Practical Techniques to Reduce Skew and Its Variations in Buffered Clock Networks*

Ganesh Venkataraman, Nikhil Jayakumar, Jiang Hu, Peng Li, Sunil Khatri
Dept. of Electrical Engineering, Texas A&M University, College Station, TX 77843
{ganesh, nikhil, jianghu, pli, sunil}@ee.tamu.edu
Anand Rajaram, Texas Instruments, Dallas, TX 75243
Patrick McGuinness, Freescale, Austin, TX 78729
Charles Alpert, IBM, Austin, TX 78758

ABSTRACT

Clock skew is becoming increasingly difficult to control due to variations. Link based non-tree clock distribution is a cost-effective technique for reducing clock skew variations. However, previous works based on this technique were limited to unbuffered clock networks and neglected spatial correlations in the experimental validation. In this work, we overcome these shortcomings and make the link based non-tree approach feasible for realistic designs. The short circuit risk and multi-driver delay issues in buffered non-tree clock networks are investigated. Our approach is validated with SPICE based Monte Carlo simulations, considering spatial correlations among variations. The experimental results show that our approach can reduce the maximal skew by 47%, improve the skew yield from 15% to 73% on average with a decrease on the total wire and buffer capacitance.

1. INTRODUCTION

In the current era of nanometer VLSI technology, process variations tend to have a significant negative impact on the performance of clock distribution networks. Non-tree clock distribution network [1–4] has been recognized as a promising approach for reducing clock skew variations, since multiple signal paths can compensate each other's variation. Among the non-tree techniques, the clock mesh [2] is perhaps the most effective and well-known one. However, mesh usually entails a large wire/power overhead and is therefore affordable only in high performance products such as microprocessors. In contrast, the link based non-tree method [3–5] is much more cost effective and will be the focus of study in this paper. Despite the advantages, previous works on link based non-tree clock network [4, 5] have a few shortcomings which hamper their applicability. The major weakness is that these works [4, 5] are limited to unbuffered clock networks. In reality, buffers are essential for clock networks and achieving desired clock skew (under a higher order delay model) in buffered cases is much more challenging. The major goal of this work is to overcome this shortcoming and make the link based non-tree technique applicable in practice. The main contributions of this work are summarized as follows.

- Link insertion in a buffered network may result in multiple drivers for a subnet. We suggest a design criterion for avoiding short circuit risk in a multi-driver net. Some analysis results obtained in [4] for single driver nets are extended for multi-driver nets.
- Skew tuning is used to synthesize a clock tree with low nominal skew under a higher order delay model. The effect of link insertion depends on a well-designed buffered clock tree. The proposed technique can decrease nominal clock skew considerably and therefore enhances the effectiveness of link insertion.
- A complete methodology of link based buffered clock network under accurate gate and wire delay models is proposed. This methodology utilizes the buffered clock tree construction techniques which are friendly to link insertion.
- The proposed method is validated with SPICE based Monte Carlo simulations considering spatially correlated power supply variations, buffer and wire process variations. Our experiments are performed on testcases with sequential adjacency information. The maximal skew, standard deviation of skew variations and skew yield [6], which is the probability of satisfying a certain skew bound, are evaluated in the Monte Carlo simulations.

2. LINK INSERTION REVIEW

For completeness, we summarize a few important conclusions from previous work on link insertion [4]. The basic idea behind the link based non-tree clock network method is to obtain a non-tree by inserting cross links between nodes in an existing clock tree. A link can be modeled as a link resistor with a pair of link capacitors at the two ends. Adding only link capacitances to a clock tree may change the skew but does not change the tree topology. The original skew can be restored by tuning the tree as in conventional clock tree routing methods.

If a link resistor is inserted between a pair of nodes with equal nominal delay (or zero nominal skew), there is no change on nominal delay at any node in the clock network. If there is skew variation between the two end nodes of the link resistor, the magnitude of the variation is *always* scaled down by the link resistance. The effect of the scaling is strong when the link resistance is small or the nearest common ancestor node of the two end nodes is close to the root. If one end of the link is in subtree T_l and the other end is in a disjoint subtree T_r, the link resistance can reduce skew variation between any pair of nodes of T_l and T_r. However, the link resistance may worsen skew variability between nodes

*This work is supported in part by Semiconductor Research Corporation under contract number 2004-TJ-1205.

in some other circumstances (see [4]). The major guidelines for link insertions include:

- Links are always inserted between nodes with zero nominal skew.
- Links are preferentially inserted between node pairs which are close to each other and their nearest common ancestor node is close to the root in the abstract topology.
- Links need to be distributed evenly in the clock network so that their skew worsening effects can cancel each other.

3. MULTI DRIVER NETS

If cross links are inserted in a buffered clock network, it is likely that a sub-net is driven by multiple buffers or drivers. This fact causes two issues which do not exist in link insertion for unbuffered clock networks. One is the risk of short circuit between the outputs of different buffers. The other is whether or not the analysis on delay and skew in [4] is still valid in the multi-driver nets. If the signal arrival times at the inputs of two buffers driving the same sub-net are significantly different, there is a risk of short circuit power consumption. Consider the example in Figure 1 where the outputs of the two buffers are initially low and then switch to high with time difference of Δ. There is a time interval Δ during which the output of upper buffer is high while the output of the lower buffer is low. Therefore, there could be a short circuit current flowing from the power supply to the ground through the upper buffer and then the lower buffer as indicated by the dashed line in Figure 1. However, the signal propagates with a finite time delay from one buffer to another. If this delay is greater than Δ, then there is not enough time to establish the short circuit current.

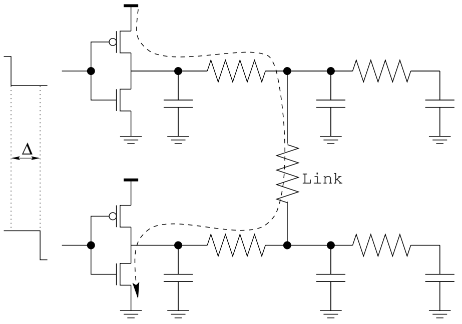

Figure 1: If there is significant difference Δ between signal arrival time to the two drivers, there is risk of short circuit indicated by the dashed line.

Denote the two buffers as B_i and B_j. Let the upper bound of the difference between signal arrival time to B_i and B_j be $\Delta_{ij,max}$ considering variations. The lower bound $\tau_{i\rightsquigarrow j}$ of signal propagation delay from B_i to B_j can be obtained through the method suggested in [7].

$$\tau_{i\rightsquigarrow j} = \frac{\sum_{(u,v)\in path(B_i\rightsquigarrow B_j)} R_{uv}^2 C_v}{\sum_{(u,v)\in path(B_i\rightsquigarrow B_j)} R_{uv}} \qquad (1)$$

where (u,v) indicates two end nodes of an edge, R_{uv} is the edge resistance and C_v is the total capacitance downstream of node v. The lower bound $\tau_{j\rightsquigarrow i}$ of signal propagation delay from B_j to B_i can be obtained similarly. Then, the criterion for avoiding short circuit between B_i and B_j is:

$$\min(\tau_{i\rightsquigarrow j}, \tau_{j\rightsquigarrow i}) > \alpha\Delta_{ij,max} \qquad (2)$$

where $\alpha > 1$ is a constant used for added safety margin.

In [4], it was shown that a link resistor inserted between two nodes with equal nominal delay always reduces the skew or the skew variation between them. However, this conclusion is for the single driver case. We will show that a multi-driver net can be converted to an equivalent single driver net and therefore the conclusion in [4] still holds for multi-driver nets. Consider the multi-driver net depicted in Figure 2(a). Let $t_1(t_2)$ denote the signal arrival time at node 1(2). Without loss of generality, let $t_1 = t_2 + \Delta$, where $\Delta \geq 0$. Consider inserting a virtual resistance R_v between the signal source s_1 and node 1 such that the delay across this virtual resistance is Δ. In such a scenario, it is easy to see that the circuit can be transformed to an equivalent single driver model shown in Figure 2(b). With the above transformation, the analysis detailed in [4] still holds good. Since inserting a link between two equal delay nodes does not affect the delay at any node, the delay across R_v can be obtained by ripping up all of the link resistance and finding the Elmore delay in the resulting tree. Therefore, the value of R_v is equal to $\frac{\Delta}{C_1}$ where C_1 is the total downstream capacitance at node 1 for the tree.

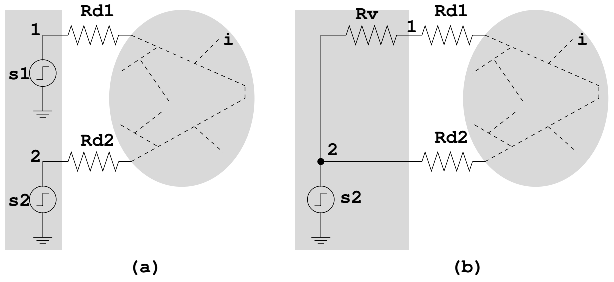

Figure 2: The dual driver net in (a) can be converted to the single driver net in (b) when signal departure time t_1 at node 1 is no less than the signal departure time t_2 at node 2.

4. LOCALIZED SKEW TUNING

For link insertion to be effective, we need to insert links between nodes with zero nominal skew [4]. However, generation of a low nominal skew tree (under a higher order delay model) is non-trivial. We will illustrate this difficulty through an example of zero skew clock tree construction in Figure 3. If zero skew has already been obtained for the

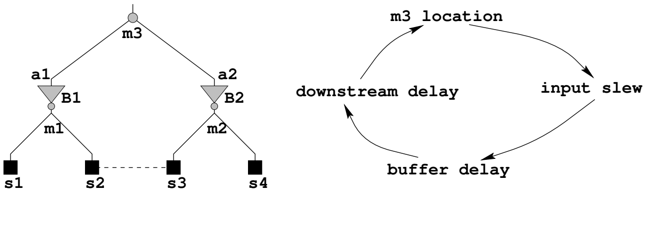

Figure 3: Tuning the location of merging node m3 in a buffered clock tree falls into a cyclic dependency.

buffered subtrees rooted at a1 and a2, we attempt to tune the location of the merging node m3 such that the delay from m3 to each sink (s1, s2, s3 or s4) is the same. Let *downstream delay* at a node k, or the delay from k to sinks, be denoted as d_k. The location of m3 is decided based on downstream delay $d_{a1} = delay(B1) + d_{m1}$ and $d_{a2} = delay(B2) + d_{m2}$. The buffer delays $delay(B1)$ and $delay(B2)$ depend on their input slew rates. However, the input slew rates depend on the location of merging node m3. We thus get into a *vicious cycle* that makes it very difficult to accurately find a merging node location that gives zero skew. This cycle is depicted at the right part of Figure 3.

The weakest link in this cycle is the dependence of merging node location on the downstream delay d_{a1} and d_{a2}. Tunable delay elements were discussed in [18] as a technique used to improve the post-silicon yield. We employ this technique to break the weakest link as well as the vicious cycle. If buffer delay $delay(B1)$ and $delay(B2)$ can be tuned without affecting other delay or slew in the buffered tree, we can decide the location of m3 regardless downstream delay d_{a1} and d_{a2} and then obtain zero skew at sinks by tuning the buffer delays. Figure 4 shows an example of a tunable buffer containing three cascaded inverters even though different number of inverters can be employed. There is a tunable dummy capacitor C between inverter I1 and inverter I2. For a given input slew and a given output load, the delay of the buffer can be tuned by sizing the dummy capacitor. Since the dummy capacitor is sandwiched between inverters in the buffer, changing its size does not affect any other delay or slew in the buffered tree but the buffer delay itself. In contrast to post-silicon tuning in [18], the tuning in our case is performed during the clock network layout and therefore does not involve any testing cost.

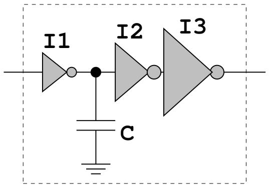

Figure 4: Tunable clock buffer.

5. LINK BASED BUFFERED CLOCK NET WORK CONSTRUCTION

Link based non-tree clock network requires a buffered tree as input. We integrate some known techniques on clock routing [10–12] with our skew tuning technique described in Section 4 to facilitate better solution quality of link insertions. An important ingredient is a balanced tree structure as illustrated in Figure 5. Such structure itself has certain tolerance to inter-die variations [10, 11] and is friendly to link insertion. The buffered tree is constructed through a bottom-up process with the topology determined by the nearest neighbor graph [9]. In order to maintain the tree balance, we impose an extra restriction that only subtrees with fewer levels are merged first. Buffers are inserted at every internal node at the same level as in Figure 5 such that the maximum load of each buffer/driver is limited.

In addition to the structural balance, we perform **delay balancing** [10] for subtrees at each level. In delay balancing, we make the delay of subtrees at the same level identical. For example, $delay(l \rightsquigarrow u) = delay(r \rightsquigarrow w)$ for Figure 5. The delay balancing can be achieved by using the tunable buffer introduced in Section 4 and sizing the dummy capacitors. Delay balancing serves two purposes. First, it avoids the risk of short circuit current. Further, links are inserted only between nodes with zero nominal skew. Without delay balancing there is no guarantee that node pairs at higher levels will have zero nominal skew.

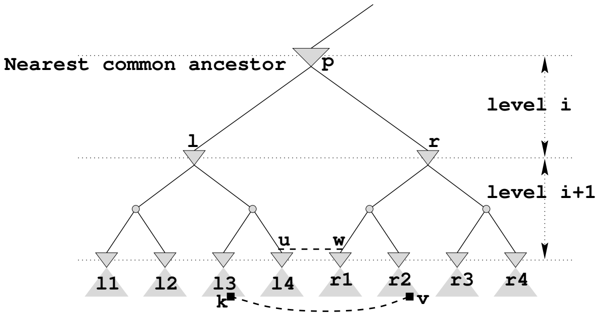

Figure 5: Link insertion in buffered clock tree. Dashed lines indicate links.

After the buffered clock tree is constructed, a SPICE simulation is performed to obtain a precise estimation on clock skew. Usually, the skew within a subtree rooted at a buffer/driver is negligible. However, there could be significant skew between different subtrees at the same level. Thus, we perform a post-processing of delay balancing through tuning the clock buffers based on SPICE simulations. For nodes driven by different buffers, we add suitable dummy capacitances to the tunable buffers to minimize skew.

The algorithm for link insertion in buffered clock networks has some significant differences from the unbuffered case [4, 5], although they share the same top-level framework below.

1. Select the node pairs for link insertion.
2. Add link capacitance to the selected nodes and perform skew tuning to restore the original skew. The skew tuning includes two steps. First, the locations of merging nodes in each subtree rooted at a buffer/driver are tuned to restore zero skew for the subtree. Next, SPICE simulation is performed to obtain a precise inter-subtree skew estimation. And the inter-subtree skew is minimized by sizing the dummy capacitors in the tunable clock buffers. This step is the same as the post-processing in the initial buffered clock tree construction.
3. Insert link resistance into the selected node pairs. Since we always select node pairs with zero nominal skew and restore such zero skew in skew tuning of previous step, the link resistances do not affect nominal skew.

Figure 6 gives the algorithm for selection of node pairs. (In Figure 6 sequentially adjacent sinks refer to those sink pairs that have only combinational logic between them and hence exhibit skew constraints). The node pair selection procedure at each sink/buffer level is derived from the MST (Minimum Spanning Tree) based algorithm in [5]. However, our algorithm differs from [5] by the fact that we employ delay balancing at levels where buffers are inserted. Such a

```
Procedure: $NodePairsBetweenTrees(T_l, T_r, m)$
Input: Two subtree $T_l$ and $T_r$,
       size indicator $m$ for each sink/buffer level
Output: A set $P$ of node pairs

$P \leftarrow \emptyset$
For each sink/buffer level deeper than $l$ and $r$
  Decompose $T_l$ to sub-subtrees $S_l = \{l_1, l_2 ... l_m\}$
  Decompose $T_r$ to sub-subtrees $S_r = \{r_1, r_2 ... r_m\}$
  Construct bipartite graph $G_{l,r}$ between $S_l$ and $S_r$
  $G_p \leftarrow$ MST of $G_{l,r}$
  For each edge $(l_i, r_j)$ in $G_p$
    If link between $l_i$ and $r_j$ has short circuit risk or
    no sequentially adjacent sinks between $l_i$ and $r_j$
      Remove $(l_i, r_j)$ from $G_p$
    Else if $degree(l_i) \neq degree(r_j)$
    or $weight(l_i, r_j) >$ threshold
      Remove $(l_i, r_j)$ from $G_p$
  $P \leftarrow P \cup$ edges in $G_p$
Return $P$
```

Figure 6: Algorithm of selecting node pairs between two subtrees.

balancing creates several pairs of zero skew nodes where a link could potentially be inserted. Moreover, our algorithm takes sequential adjacency into account.

6. EXPERIMENTAL RESULTS

The ISCAS89 benchmark suite is employed for our experiments as it includes relatively complete circuits with both combinational and sequential elements so that the sequential adjacency information for clock sinks (flip-flops) are available. The experiments were designed to test the effect of delay balancing as well as link insertion on both nominal skew and skew due to process variations.

First, synthesis is performed on the ISCAS89 benchmark circuits by using SIS [13] to obtain a mapped netlist. Then, an academic placement tool *mPL* [14] is employed to get circuit placement. Assuming a $180nm$ technology, wire capacitance values are obtained by using the SPACE 3D extraction tool [16]. The other wire parameters such as ILD (inter-layer dielectric) dimensions and sheet resistances are taken from [17]. Gate/buffer models and timing simulations are based on HSPICE with $180nm$ technology parameters from [15].

Our techniques are implemented in C++ and run on a Sun Solaris Ultra Sparc machine with 2GB RAM. Clock skew variations are evaluated through Monte Carlo simulations. Variations on buffer channel length, power supply level, wire width and sink load capacitance are considered. These variations are assumed to follow Gaussian distribution with standard deviations equal to 5% of their nominal values. Spatial correlations among the variations are handled by the PCA (Principle Component Analysis) method as in [19]. The experiments are designed to test the effect of the proposed techniques:

- **Balancing vs. non-balancing.** The tunable buffer (Section 4) based delay balancing (Section 5) is a main technique for both tuning nominal skew and facilitating link insertion. Thus, both non-balancing and balancing cases are tested in the experiments.

	Balancing + link				
Case	Delay	Skew	WL	Total cap	CPU
s9234	370	0.8	39866	6.96	32
s5378	380	0.7	43097	7.62	53
s13207	674	11.8	130074	23.13	413
Ave	475	4.4	71012	12.6	166

Table 2: Nominal results for link based buffered clock networks. The maximal source-sink delay and the maximal skew are in ps. The wirelength (WL) results are in μm. The total capacitance is in pF. CPU time is in seconds.

- **Link vs. WO-link.** Link insertion is the core technique of the proposed approach. Results with link insertion are compared with those without link insertion (WO-link).

The experimental results are organized in two parts:

- **Nominal results.** In this part, the maximal source-sink delay (ps) and the maximal skew (ps) are reported. In addition, we show the major resource usages including wirelength (**WL**) in μm, number of buffers and the CPU time of running our algorithms. We also report total wire and buffer capacitance which includes the tunable dummy capacitances so that there is an overall estimation on wire and buffer cost on a normalized basis.
- **Variation results.** These results are obtained from 1000 iterations of Monte Carlo simulations for each case. The maximum skew (**MS**) and standard deviation (**SD**) of skew among all iterations are recorded. In addition, the skew yield (**SY**) [6], which is the probability of satisfying a certain skew bound (**SB**), is reported.

The nominal results without (with) link insertions are listed in Table 1 (Table 2). These data show that our delay balancing technique can reduce both source-sink delay and the nominal skew significantly. As expected, link insertion has small influence on the nominal delay and skew. Since the delay balancing is based on sizing the dummy capacitors within tunable buffers, we need to observe increase in the buffer capacitance including the dummy capacitance. The capacitance data indicates that the total buffer capacitance is much smaller than the total wire capacitance. Therefore, the buffer capacitance increase is often smaller than the wire capacitance reduction due to delay balancing. The CPU time for delay balancing and the skew tuning for link insertion tends to be high as implied by the rightmost column of Table 1 and Table 2. This is mostly due to the SPICE runtime required to size the dummy capacitors.

The variation results are shown in Table 3. The results were computed using Monte Carlo simulations considering spatial correlations. Skew yield refers to the percentage number of trials where the maximum skew is less than the specified skew bound (SB). One can see that the delay balancing technique can actually reduce skew variation in terms of the improvement on the maximal skew (MS) and the skew yield (SY). Obviously, the link insertion can improve the standard deviation (SD) in addition to the maximum skew and skew yield. The average skew yield is improved from 15% to 73% by our method.

		Non-balancing, WO-link						Balancing, WO-link				
Case	#Sinks	Delay	Skew	WL	#Buf	Total cap	CPU(s)	Delay	Skew	WL	Total cap	CPU
s9234	135	468	97	41498	8	7.24	0.3	367	0.5	37043	6.44	31
s5378	164	612	71	46218	20	8.09	1.0	379	2.9	42522	7.41	51
s13207	500	663	60	133650	80	23.15	2.0	662	11.7	129203	23.01	203
Ave	266	581	76	73789	36	12.8	1.1	469	5.0	69589	10.8	95

Table 1: Nominal results for buffered clock trees. The maximal source-sink delay and the maximal skew are in ps. The wirelength (WL) results are in μm. The total capacitance is in pF. CPU time is in seconds.

	Non-balancing, WO-link				Balancing, WO-link			Balancing + link		
Case	MS	SD	SB	SY	MS	SD	SY	MS	SD	SY
s9234	175	23	50ps	0%	112	16	42%	64	10	95%
s5378	217	32	50ps	5%	95	13	38%	94	14	63%
s13207	247	33	100ps	41%	243	33	48%	180	27	61%
Norm ave	1	1		1	0.70	0.72	2.87	0.53	0.59	4.87

Table 3: Variation results. The maximal skew (MS), standard deviation (SD) and skew bound (SB) are in ps. The last row shows the normalized average values.

7. CONCLUSIONS

In order to cope with the increasingly significant skew variations, we propose a link insertion technique for buffered clock networks. The induced multi-driver issues such as short circuit risk and multi-driver delay estimation are studied. Experimental results from SPICE based Monte Carlo simulations confirm the effectiveness of our approach.

8. ACKNOWLEDGMENT

The authors would like to thank Dr. Xiaodong Yang for some helpful discussions.

9. REFERENCES

[1] N. A. Kurd, J. S. Barkatullah, R. O. Dizon, T. D. Fletcher, and P. D. Madland. A multigigahertz clocking scheme for the Pentium 4 microprocessor. *IEEE Journal of Solid-State Circuits*, 36(11):1647–1653, November 2001.

[2] P. J. Restle, T. G. McNamara, D. A. Webber, P. J. Camporese, K. F. Eng, K. A. Jenkins, D. H. Allen, M. J. Rohn, M. P. Quaranta, D. W. Boerstler, C. J. Alpert, C. A. Carter, R. N. Bailey, J. G. Petrovick, B. L. Krauter, and B. D. McCredie. A clock distribution network for microprocessors. *IEEE Journal of Solid-State Circuits*, 36(5):792–799, May 2001.

[3] N. Bindal, T. Kelly, N. Velastegui, and K. L. Wong. Scalable sub-10ps skew global clock distribution for a 90nm multi-GHz IA microprocessor. In *Proceedings of the IEEE International Solid-State Circuits Conference*, pages 346–355, 2003.

[4] A. Rajaram, J. Hu, and R. Mahapatra. Reducing clock skew variability via cross links. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 18–23, 2004.

[5] A. Rajaram, D. Pan, and J. Hu. Improved algorithms for link based non-tree clock network for skew variability reduction. In *Proceedings of the ACM International Symposium on Physical Design*, pages 55–62, 2005.

[6] X. Jiang and S. Horiguchi. Statistical skew modeling for general clock distribution networks in presence of process variations. *IEEE Transactions on VLSI Systems*, 9(5):704–717, October 2001.

[7] J. Rubinstein, P. Penfield, and M. A. Horowitz. Signal delay in RC tree networks. *IEEE Transactions on Computer-Aided Design*, CAD-2(3):202–211, July 1983.

[8] T.-H. Chao, Y.-C. Hsu, J.-M. Ho, K. D. Boese, and A. B. Kahng. Zero skew clock routing with minimum wirelength. *IEEE Transactions on Circuits and Systems - Analog and Digital Signal Processing*, 39(11):799–814, November 1992.

[9] M. Edahiro. A clustering-based optimization algorithm in zero-skew routings. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 612–616, 1993.

[10] S. Pullela, N. Menezes, J. Omar, and L. T. Pillage. Skew and delay optimization for reliable buffered clock trees. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 556–562, 1993.

[11] J. G. Xi and W. W.-M. Dai. Buffer insertion and sizing under process variations for low power clock distribution. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 491–496, 1995.

[12] R. Chaturvedi and J. Hu. Buffered clock tree for high quality IC design. In *Proceedings of the IEEE International Symposium on Quality Electronic Design*, pages 381–386, 2004.

[13] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. L. Sangiovanni-Vincentelli. SIS: a system for sequential circuit synthesis. Memorandum no. M92/41, ERL, University of California, Berkeley, May 1992.

[14] CPMO-constrained placement by multilevel optimization. http://ballade.cs.ucla.edu/cpmo/. Computer Science Department, UCLA.

[15] Y. Cao, T. Sato, D. Sylvester, M. Orshansky, and C. Hu. New paradigm of predictive MOSFET and interconnect modeling for early circuit simulation. In *Proceedings of the IEEE Custom Integrated Circuits Conference*, pages 201–204, 2000.

[16] SPACE: VLSI physical design modeling and verification. http://space.tudelft.nl. Delft University of Technology in the Netherlands.

[17] S. P. Khatri, A. Mehrotra, R. K. Brayton, A. Sangiovanni-Vincentelli, and R. H. J. M. Otten. A novel VLSI layout fabric for deep sub-micron applications. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 491–496, 1999.

[18] J. L. Tsai, D. Baik, C. C. Chen and K. Saluja A yield improvement methodology using pre- and post-silicon statistical clock scheduling In *Proceedings of the IEEE/ACM International Conference on Computer Aided Design*, 2004.

[19] H. Chang and S. S. Sapatnekar. Statistical timing analysis considering spatial correlations using a single PERT-like traversal. In *Proceedings of the IEEE/ACM Int Conference on Comp. Aided Design* , pages 621–625, 2003.