Process-Variation Robust and Low-Power Zero-Skew Buffered Clock-Tree Synthesis Using Projected Scan-Line Sampling

Abstract

Process-variation induced skew has become one of the major contributors to the clock-skew in advanced technologies. Since process-variation induced skew is roughly proportional to clock-delay, it is preferable to design zero-skew clock-trees and have minimum clock-delay to reduce both unintentional and process-variation induced skews. In this paper, we propose a zero-skew buffered clock-tree synthesis flow and a novel algorithm that enables clock-tree optimization throughout the full zero-skew design-space by considering simultaneous buffer-insertion, buffer-sizing, and wire-sizing. Our algorithm can also utilize useful-skew and bounded-skew constraints to allow more aggrressive optimization. Extensive experimental results show that for an industrial clock-tree with 3101 sink nodes, our algorithm achieves up to 45X clock-delay improvement and up to 23% power reduction compared with its initial routing and provides process-variation, clock-delay, and power trade-offs in 16 minutes on a 1.2GHz Pentium IV PC.

1 Introduction

In the multi-giga-Hertz design era, clock design plays a crucial role in determining chip performance and facilitating timing and design convergence. Since clock-skew directly translates to cycle-time penalty, zero-skew clock-tree designs have received significant attention. Various approaches for zero-skew clock-tree construction have been proposed [13] [2] [8]. As technology feature size shrinks, process-variation induced clock-skew dominates the clock-skew in manufactured chips. Process-variation comes from various manufacturing procedures such as lithography, etching, chemical-mechanical polishing, etc., and is usually difficult to predict. It is observed in [7] that process-variation induced skew can reach 10% of the clock-delay.

The total clock-delay is dominated by interconnect-delay in a clock-tree. To reduce such delay, buffer-insertion, buffer-sizing, and wire-sizing have been widely adopted. These techniques not only improve the robustness of a clock-tree but also reduce power-consumption by avoiding the use of wide wires and thereby reducing clock-tree capacitance. An excellent survey of interconnect optimization techniques can be found in [5]. Buffer-insertion, buffer-sizing, and wire-sizing techniques for delay and power optimization have been proposed in two major categories. In the first category, buffer-insertion, buffer-sizing, and wire-sizing are formulated as optimization problems, in which the maximum delay at each leaf node is constrained [6] [4] [3]. The second category, based on van Ginneken's algorithm, uses a bottom-up dynamic programming technique to find optimal solutions for a subtree and propagate the solutions up toward the root node [8] [14] [9] [1].

Recent work [12] uses a new approach to tackle wire-sizing problems for zero-skew clock-tree optimization. Since a zero-skew clock-tree can be characterized by its clock-delay and capacitance, the proposed algorithm captures the zero-skew design-space of each subtree with a two dimensional region, or a *DC region*, where the coordinates of the Y-axis and the X-axis are the clock-delay and capacitance values. Every point inside a DC region represents one or more zero-skew designs, or *embeddings*, of the subtree. Sampling techniques are used to capture the irregular shapes of the DC regions. The DC region of a clock-tree is constructed by the sampled subtree DC regions in a bottom-up fashion. A top-down embedding selection algorithm is then used to generate the embeddings that yield the chosen clock-delay and capacitance values from the DC regions. The major advantages of the proposed algorithm are that it finds ϵ-*optimal* solutions and takes pseudo-polynomial runtime and memory usage. However, it does not consider buffer-insertion and buffer-sizing that are more efficient in reducing clock-delays.

In this paper, we extend the work of [12] and propose a zero-skew buffered clock-tree synthesis flow that generates process-variation robust and low-power clock-trees from a given set of clock sink nodes. First, we adopt the BB+DME [2] algorithm to generate the initial routings. The initial routings are then optimized for process-variation and power with a novel algorithm that performs simultaneous buffer-insertion/sizing and wire-sizing within the zero-skew design-space. The zero-skew design-space of each buffered subtree is represented by a three dimensional DC region and sampling techniques are extended to capture it. In order to handle inverter-insertion, which is a more area efficient alternative to buffer-insertion [11], and the signal-polarity issue, two sets of DC regions are used. After the bottom-up DC region construction, a top-down embedding selection algorithm is used to generate embeddings from the DC regions.

The rest of this paper is organized as follows: in Section 2, the delay and power models and the DC region approach are discussed. Section 3 introduces the proposed clock-tree synthesis flow and simultaneous buffer-insertion/sizing and wire-sizing algorithm. The top-down embedding selection algorithm is proposed in Section 4. Experimental results are discussed in Section 5 and finally Section 6 concludes this work.

2 Preliminaries

2.1 Delay and Power Models

Interconnect-delay and gate-delay are two delay components in a clock-tree. In this paper, interconnects and buffers are modeled with the resistance-capacitance (RC) model with delays according to the Elmore delay model. A clock-tree with given routing rooted at node v is denoted T_v. For a wire with length l and width w, the wire resistance is $\frac{lr_0}{w}$ and the wire capacitance is lc_0w, where r_0 and c_0 are the resistance and capacitance of a $1\mu m^2$ wire. The wire capacitance is modeled with two equal capacitors attached to both ends of the wire. For a buffer with gate width w_b, the gate capacitance at its input is w_bc_b and its effective output resistance is $\frac{r_b}{w_b}$, where c_b and r_b are unit-width gate capacitance and resistance. The gate is modeled as a ramp voltage source with an intrinsic delay of t_c.

The clock-tree power consumption is generally modeled as $P = fCV^2 + P_s + P_l$, where f is the switching frequency, V is the voltage swing, C is the sum of all interconnect capacitance, buffer gate capacitance, and sink loads. P_s accounts for the buffer short-circuit power and P_l accounts for the leakage power. Since P_s and P_l are usually much smaller compared to fCV^2, which alone is a reasonably accurate measure of the total power consumption in clock-tree designs [10], we adopt the simplified model $P = fCV^2$ in this paper.

2.2 The DC Region Approach

In the Elmore delay model each subtree is characterized by the total capacitance seen at its root node. For a clock-tree T_v, each embedding determines a pair of clock-delay and capacitance values (d_v, c_v). Thus the whole zero-skew design-space of T_v can be represented by the DC region Ω_v on the DC plane, an X-Y plane where the coordinates of the Y-axis and the X-axis are the clock-delay and capacitance values. Figure 1 shows the DC regions of T_v, T_{v_l},

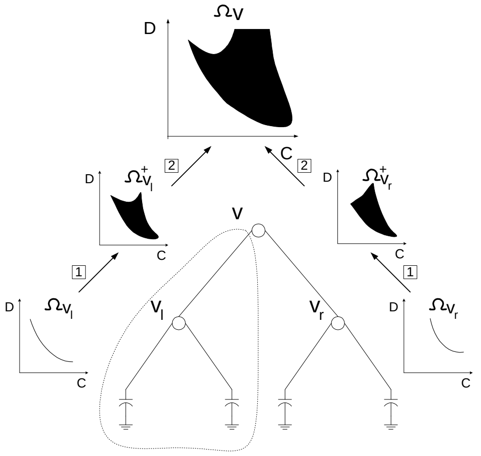

Figure 1: The DC regions of the subtrees in a clock-tree T_v.

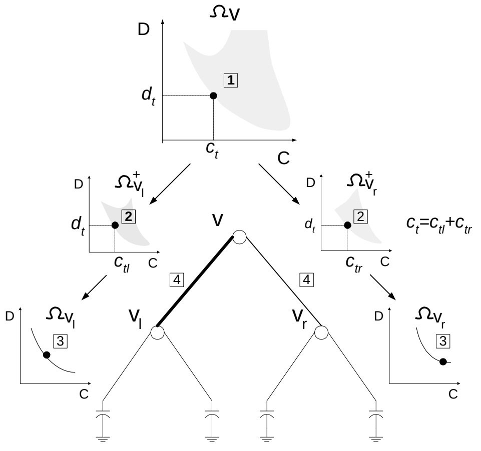

Figure 2: Illustration of the embedding selection steps.

and T_{v_r} considering only wire-sizing. The DC region of a level-1 node such as Ω_{v_l} is a curve and the DC regions of higher level nodes have complex shapes. From the DC regions we can learn the minimum-delay and minimum-power achieved by wire-sizing. The Y-coordinate of the bottom-most point in a DC region is the minimum-delay and the X-coordinate of the left-most point in a DC region is the minimum-power.

A clock-tree is composed of two *branches*. The left branch of T_v is enclosed in the dashed circle in Figure 1 and is denoted $T^+_{v_l}$. The DC region of $T^+_{v_l}$, a *branch DC region*, is denoted $\Omega^+_{v_l}$ and it can be obtained by applying a transformation on Ω_{v_l}. The zero-skew constraint requires the delays along both branches to be the same, thus Ω_v can be obtained by an equi-delay merge operation on $\Omega^+_{v_l}$ and $\Omega^+_{v_r}$. Thus the DC region of a clock-tree can be constructed in a bottom-up fashion recursively with two major steps. First the branch DC regions are obtained by transforming the DC regions of the left and right subtrees. Second, the DC region is obtained by merging the branch DC regions.

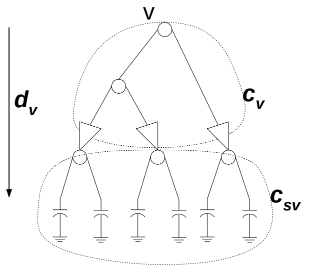

Figure 3: Characterize a buffered clock-tree T_v with three parameters.

Generating embeddings from the DC regions is an inverse process. First the target clock-delay and capacitance, d_t and c_t, of an embedding are chosen. The capacitance is then split into two branches. The DC regions of the left and right subtrees are scanned and feasible target clock-delays and capacitances of the subtrees are selected. From the selected points in the branch DC regions and the subtree DC regions the wire widths above the subtrees can be determined. The process is illustrated in Figure 2.

3 Simultaneous Buffer-Insertion/Sizing and Wire-Sizing

In our zero-skew buffered clock-tree synthesis flow the BB+DME [2] algorithm is used to generate initial routings. A set of clock sink nodes is first recursively bi-partitioned according to the locations and the capacitances of the sinks such that the two partitioned subsets have balanced number of sink nodes and total capacitances. The balanced bi-partition continues until each subset contains only one sink node and then an abstract binary-tree topology is obtained. The *merging segments*, sets of zero-skew merging points, for each internal node are constructed in a bottom-up fashion. Finally a set of merging points are selected in a top-down fashion and an initial clock-tree is generated. We consider buffer-insertion above all merging points and obtain a buffered clock-tree.

A buffered clock-tree T_v can be characterized by three parameters as shown in Figure 3. The first two parameters, d_v and c_v, are the clock-delay and the capacitance seen at the root node v. The total switching capacitance of T_v is the sum of c_v and c_{sv}, the capacitance shielded from v by the buffers. Thus we can represent the zero-skew design-space with a three dimensional DC region, where the coordinates of the Z-axis, X-axis, and Y-axis are the clock-delay, capacitance seen at the root node, and the shielded capacitance.

Significant extensions are done to handle the three dimensional DC regions. Due to the space limitation we only highlight the important extensions in the following subsections.

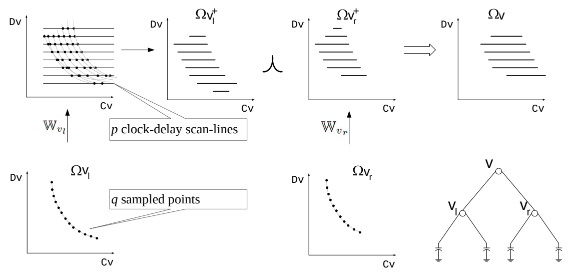

Figure 4: Obtain the sampled DC region of an unbufferd two-level clock-tree.

3.1 Branch DC Regions and Projected Scan-Line Sampling

The details of transforming subtree DC regions to a branch DC region for wire-sizing is available in [12]. To capture the irregular shapes of the branch DC regions, sampling on wire-width is applied and the intersection of the branch DC regions and a set of clock-delay scan-lines are obtained. The branch DC regions can then be represented by a set of horizontal segments. Since the branch DC region is the projection of a zero-skew design-space, we refer the project-sample-scan procedure as *projected scan-line sampling*. The transformation is defined by the following equations obtained directly from the Elmore delay model:

$$d_v^+ = d_v + \frac{l_v r_0}{w_e(v)}\left(\frac{l_v w_e(v) c_0}{2} + c_v\right) \tag{1}$$

$$c_v^+ = c_v + l_v w_e(v) c_0, \tag{2}$$

$$\forall\, (d_v, c_v) \in \Omega_v.$$

The edge connecting v to its parent node is e_v and the length of e_v is l_v. The wire-width of e_v is $w_e(v)$ and $w_m \leq w_e(v) \leq w_M$. Since wire-sizing does not affect c_{sv} we can set $c_{sv}^+ = c_{sv}$ and use projected scan-line sampling to obtain three dimensional sampled branch DC regions. It is denoted

$$\Omega_v^+ = \mathbb{W}_v(\Omega_v).$$

If a buffer is inserted above a subtree, then the transformation is defined by the following equations:

$$d_v^+ = d_v + t_c + \frac{l_v^2 r_0 c_0}{2} + \frac{l_v r_0 c_b w_b(v)}{w_e(v)} + \frac{r_b c_v}{w_b(v)} \tag{3}$$

$$c_v^+ = c_b w_b(v) + l_v w_e(v) c_0, \tag{4}$$

$$c_{sv}^+ = c_{sv} + c_v, \tag{5}$$

$$\forall\, (d_v, c_v, c_{sv}) \in \Omega_v.$$

Equation (3) defines the difference of clock-delays between a branch and its subtree, which is composed of the buffer intrinsic delay, the delay caused by wire and buffer capacitance, and the buffer-delay by driving the shielded capacitance. The width of the buffer above v is $w_b(v)$. Equation (4) (5) define the change of the capacitance seen from

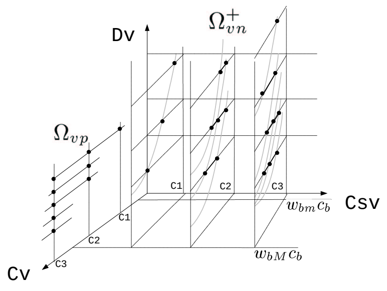

Figure 5: Obtain the sampled branch DC region of a buffered branch.

the root and the shielded capacitance of a branch. Samplings are applied on buffer-width and shielded capacitance, and projected scan-line sampling is used to obtain the sampled DC regions with a set of horizontal segments along the X-axis. It is denoted

$$\Omega_v^+ = \mathbb{B}_v(\Omega_v).$$

The equi-delay merge takes one segment from each sampled branch DC region with the same clock-delay and generate a new segment where $c_v = c_{v_l} + c_{v_r}$ and $c_{sv} = c^+_{sv_l} + c^+_{sv_r}$. The merge operator is denoted $\curlywedge$ and the merge operation is written as

$$\Omega_v \leftarrow \Omega^+_{v_l} \curlywedge \Omega^+_{v_r}.$$

Figure 4 shows the steps of projected scan-line sampling and equi-delay merge in obtaining the sampled DC region of an unbuffered two-level clock-tree.

3.2 Inverter-Insertion

In practice, inverter-insertion uses less area and is preferred over buffer-insertion. With an inverter inserted, the clock-phase is inverted. This is accommodated in our model by keeping two DC regions and branch DC regions at each node v, where

$$\Omega_v = \Omega_{vp} \cup \Omega_{vn},$$
$$\Omega_v^+ = \Omega^+_{vp} \cup \Omega^+_{vn}.$$

From this point on in this paper the term buffer refers to inverter. Figure 5 shows the steps of projected scan-line sampling in obtaining the sampled branch DC region from the sampled DC region. The complete bottom-up DC region construction algorithm is presented in Algorithm 1.

3.3 Complexity

Assume a clock-tree has n nodes and the numbers of samples used for clock-delay scan-line, buffer-width, and shielded capacitance are p, q, and r. A sampled DC region is represented by r groups of horizontal segments lying on X-Z planes, with each group containing p segments. To generate a branch DC region, q curves are generated from each segment, and $p \cdot q \cdot r$ curves are used to intersect with the $p \cdot r$ scan-lines to generate another r groups of segments. Thus the complexity of our algorithm is $O(np^2qr^2)$ and memory usage is $O(npr)$. By exploring the properties of (3) (4) (5), runtime can be reduced to $\sim O(npqr^2)$. Note that there can be more than one segments on a clock-delay scan-line. However, the gaps between the segments tend to be filled-up quickly as we move upward toward the root node. For example, multiple segments can overlap and become a single segment either when we create the sampled branch DC regions or merge the sampled branch DC regions with the $\curlywedge$ operator. In practice, the number of segments on a scan-line is always less than four and we exclude it in the complexity analyses.

Algorithm 1 *Construct_DC(T_v)*

```
Input: a clock-tree $T_v$ with given routing rooted at node $v$
Output: DC regions and branch DC regions of $T_v$
if $v$ is a leaf node then
    $\Omega_{vp} \leftarrow (d_v, c_v, 0), \Omega_{vn} \leftarrow \phi$
else {$v$ is an internal node}
    call Construct_DC($T_{v_l}$)
    call Construct_DC($T_{v_r}$)
    $\Omega^+_{v_lp} \leftarrow \mathbb{W}_{v_l}(\Omega_{v_lp}) \cup \mathbb{B}_{v_l}(\Omega_{v_ln})$
    $\Omega^+_{v_ln} \leftarrow \mathbb{W}_{v_l}(\Omega_{v_ln}) \cup \mathbb{B}_{v_l}(\Omega_{v_lp})$
    $\Omega^+_{v_rp} \leftarrow \mathbb{W}_{v_r}(\Omega_{v_rp}) \cup \mathbb{B}_{v_r}(\Omega_{v_rn})$
    $\Omega^+_{v_rn} \leftarrow \mathbb{W}_{v_r}(\Omega_{v_rn}) \cup \mathbb{B}_{v_r}(\Omega_{v_rp})$
    $\Omega_v \leftarrow (\Omega^+_{v_lp} \curlywedge \Omega^+_{v_rp}) \cup (\Omega^+_{v_ln} \curlywedge \Omega^+_{v_rn})$
end if
```

Compared with van Ginneken's algorithm based approaches, which suffer from exponential growth on the size of the solution set, our approach only takes polynomial runtime and memory usage. This advantage comes from projected scan-line sampling. Since each line segment captures an infinite number of solutions, our approach encode more information with the same amount of memory and result in better runtime and memory usage.

3.4 Slew-Rate Control and Useful-Skew

One of the purposes of buffer-insertion is to adjust the clock slew-rate. If the load capacitance of a buffer is too large, the output signal has a slow rise and fall time, which in turn increases the short-circuit power of its downstream buffers. One way to control the slew-rate is to limit the load capacitance to a certain value such that the slew-rate of the buffer is bounded to a desired value. This constraint is accounted for in our approach by limiting c_v in the bottom-up phase and discarding the portion of the design-space with long slew-rates. In the bottom-up phase, the DC regions can become large because of the embeddings with excessive buffers, which have large clock-delay and total capacitances. By setting upper limits on d_v and $(c_{sv} + c_v)$, those ill-buffered embeddings are excluded in our DC regions.

Recently, useful-skew [15] concepts have been widely

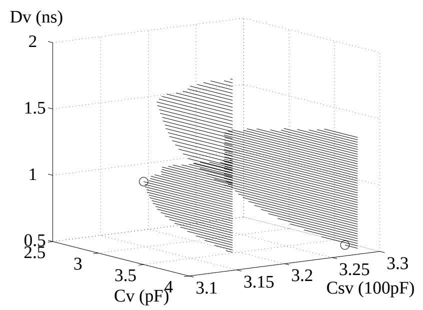

Figure 6: The sampled DC region of $r5$. The circles indicate the minimum-delay and minimum-power solutions

proposed to speed up timing convergence in order to compensate for timing uncertainties in physical layouts. Our algorithm accommodates DC region generation for a useful-skew design-space by assigning useful-skew values to d_v at each leaf node. Furthermore, in low-power designs not requiring high clock-frequencies, DC regions can be generated for a bounded-skew design-space to allow more aggressive power-optimization. In this case, the skew-bounds are assigned to d_v. Ω_v of a leaf node becomes a vertical segment, of which the Y-coordinates of the end points are the maximum and minimum acceptable skews and the X-coordinate is the sink capacitance.

4 Embedding Selection

The top-down embedding selection algorithm is illustrated by the following five steps:

1. If node v is root, select desired $(\hat{d}_v, \hat{c}_v, \hat{c}_{sv}) \in \Omega_v$.
2. With $(\hat{d}_v, \hat{c}_v, \hat{c}_{sv}) \in \Omega_v$, determine $(\hat{d}_v, \hat{c}^+_{v_l}, \hat{c}^+_{sv_l}) \in \Omega^+_{v_l}$ and $(\hat{d}_v, \hat{c}^+_{v_r}, \hat{c}^+_{sv_r}) \in \Omega^+_{v_r}$ by breaking $\hat{c}_v$ and $\hat{c}_{sv}$, respectively, into $\hat{c}^+_{v_l}$, $\hat{c}^+_{v_r}$, and $\hat{c}^+_{sv_l}$, $\hat{c}^+_{sv_r}$ such that $\hat{c}_v = \hat{c}^+_{v_l} + \hat{c}^+_{v_r}$, $\hat{c}_{sv} = \hat{c}^+_{sv_l} + \hat{c}^+_{sv_r}$.
3. With $(\hat{d}_v, \hat{c}^+_{v_l}, \hat{c}^+_{sv_l}) \in \Omega^+_{v_l}$, find $(\hat{d}_v, \hat{c}_{v_l}, \hat{c}_{sv_l}) \in \Omega_{v_l}$ such that either (1)-(2) or (3)-(5) are satisfied.
   - To satisfy (1)-(2), $b_{v_l} = \phi$, $\hat{c}_{sv_l} = \hat{c}^+_{sv_l}$, and $w_e(v_l)$ is determined by $\hat{c}_{v_l}$.
   - To satisfy (3)-(5), $\hat{c}_{v_l} = \hat{c}^+_{sv_l} - \hat{c}_{sv_l}$, and $w_b(v_l)$, $w_e(v_l)$ are determined by $(\hat{d}_{v_l}, \hat{c}_{v_l}, \hat{c}_{sv_l})$.
4. Repeat Step 3 for $(\hat{d}_v, \hat{c}^+_{v_r}, \hat{c}^+_{sv_r}) \in \Omega^+_{v_r}$.
5. Repeat the complete process at v_l and v_r.

To highlight the purpose of each step, Step 1 determines the target clock-delay, capacitance seen at the root node, and shielded capacitance, Step 2 splits the capacitance budgets to two branches T^+_{vl} and T^+_{vr}, and Step 3 determines buffer widths for b_v and wire widths for e_v.

5 Experimental Results

We implement our algorithm in C++ and run the program on a 1GB 1.2GHz Pentium IV PC. The benchmarks $r1$-$r5$ are taken from [13]. All simulations use $r_0 = 0.03\Omega$, $c_0 = 2 \times 10^{-16}F$, $w_m = 0.3\mu m$, $w_M = 3\mu m$. The parameters of the buffers are $c_b = 40fF$, $r_b = 100\Omega$, $t_c = 30ps$, $w_{bm} = 1$, $w_{bM} = 10$, and the channel length equals $0.3\mu m$. The initial routings are generated by the BB+DME [2] algorithm. The sampling resolution is set to $p = q = r = 64$.

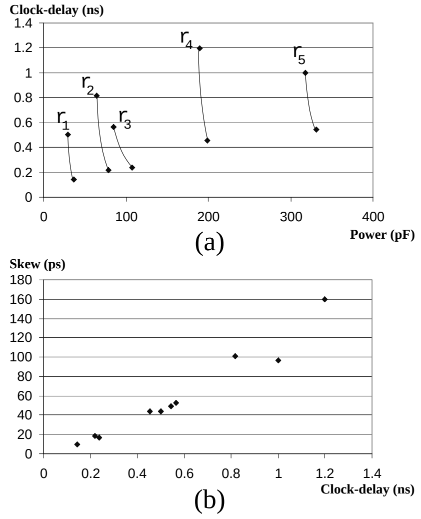

Figure 7: (a) Clock-delay and power trade-off curves and (b) Worst-case process-variation induced skews of the minimum-delay and minimum-power embeddings of $r1$-$r5$.

Table 1 shows the minimum-delay and minimum-power solutions captured by projected scan-line sampling. The delay gains are the initial delays divided by the optimized delays and the load gains are the load reductions divide by the initial loads. With simultaneous buffer-insertion, buffer-sizing, and wire-sizing, the minimum-power embeddings consume $23\% \sim 34\%$ less power and delay improves $2X \sim 24X$ than their initial routings. Minimum-delay solutions have more than 2X speedup compared with minimum-power solutions. The power difference between minimum-delay and minimum-power solutions decreases with larger circuits and drops to 5% in $r5$. Compared with wire-sizing along, which achieves an average of 3.3X delay reduction with 21.6% more power (with $1\mu m \leq w \leq 4\mu m$) [12], simultaneous buffer-insertion/sizing and wire-sizing is not only much more efficient in reducing clock-delay but also more power efficient. Figure 6 shows the DC region of $r5$.

Figure 7(a) shows the clock-delay and power trade-off curves of $r1$-$r5$. The power difference between minimum-delay and minimum-power embeddings are relatively small compared with their clock-delay difference. Figure 7(b) shows the worst-case process-variation induced skews of the minimum-delay and minimum-power embeddings of $r1$-$r5$ with four systematic process-variation models. In these models, cross-chip feature-size variation increases linearly from $0\mu m$ to $0.03\mu m$ affecting wire-widths, buffer-

Input	Initial($w = 1\mu m$)		*Construct_DC* ($w_m = 0.3\mu m, w_M = 3\mu m, w_{bm} = 1, w_{bM} = 10, p = q = r = 64$)										
	Delay	Load	Minimum-delay solution					Minimum-power solution					CPU
	(ns)	(pF)	Delay	Gain	Load	Gain	Buffers	Delay	Gain	Load	Gain	Buffers	(min.)
$r1$(267)	1.097	45.2	0.145	7.57	36.6	19.0%	34	0.500	2.20	29.8	34.1%	25	1.4
$r2$(598)	3.210	93.6	0.218	14.72	78.9	15.7%	61	0.818	3.93	65.0	30.6%	67	3.2
$r3$(862)	4.590	126.7	0.236	19.43	107.9	14.8%	106	0.563	8.15	84.6	33.2%	64	3.6
$r4$(1903)	13.184	266.2	0.454	29.02	198.6	25.4%	116	1.199	10.99	189.5	28.8%	115	10.5
$r5$(3101)	24.883	413.0	0.545	45.65	331.5	19.8%	384	0.999	24.90	317.6	23.2%	280	16.1

Table 1: Clock-delay and power consumption before and after buffer-insertion/sizing and wire-sizing. The delays shown are the Elmore-delays multiplied by $ln2$.

widths, and buffer channel-lengths. The four process-variation models are: top-to-bottom, bottom-to-top, left-to-right, and right-to-left. Not surprisingly, process-variation induced skew is highly correlated to clock-delay and Figure 7(a) is a good approximation for process-variation and power trade-offs. For low-power applications we can trade clock-delay and process-variation robustness for power by choosing minimum-power embeddings. For high performance designs minimum-delay embeddings are in general preferable for their higher process-variation tolerance and near optimal power-consumption.

6. Conclusion and Future Work

We present a novel zero-skew buffered clock-tree synthesis algorithm using projected scan-line sampling that utilizes simultaneous buffer-insertion/sizing and wire-sizing. Compared with the previous work that uses wire-sizing along, our algorithm reduces clock-delay significantly and uses less power. Experimental results show that our algorithm provides process-variation, clock-delay, and power trade-offs with efficient runtime and memory usage. In high performance designs minimum-delay embeddings are better due to their robustness to process-variation and their near optimal power-consumption.

We are currently extending our algorithm for zero-skew gated clock-tree optimization. We also plan to investigate the impacts of different clock-routings on clock-delay, power-consumption, and robustness to process-variation.

References

[1] C. J. Alpert, A. Devgan, and S. T. Quay. Buffer insertion with accurate gate and interconnect delay computation. In *Proceedings of the 36th ACM/IEEE conference on Design automation conference*, pages 479–484. ACM Press, 1999.

[2] T.-H. Chao, Y.-C. Hsu, J.-M. Ho, and A. Kahng. Zero skew clock routing with minimum wirelength. *Circuits and Systems II: Analog and Digital Signal Processing*, Volumn 39, Issue 11:799–814, Nov. 1992.

[3] C.-P. Chen, C. C. N. Chu, and D. F. Wong. Fast and exact simultaneous gate and wire sizing by lagrangian relaxation. In *Proceedings of the 1998 IEEE/ACM international conference on Computer-aided design*, pages 617–624. ACM Press, 1998.

[4] J. Cong, C. Koh, and K. Leung. Simultaneous buffer and wire sizing for performance and power optimization. In *Proceedings of the 1996 international symposium on Low power electronics and design*, pages 271–276. IEEE Press, 1996.

[5] J. Cong, Z. Pan, L. He, C.-K. Koh, and K.-Y. Khoo. Interconnect design for deep submicron ics. In *Proceedings of the 1997 IEEE/ACM international conference on Computer-aided design*, pages 478–485. IEEE Computer Society, 1997.

[6] R. Kay, G. Bucheuv, and L. T. Pileggi. Ewa: exact wiring-sizing algorithm. In *Proceedings of the 1997 international symposium on Physical design*, pages 178–185. ACM Press, 1997.

[7] S. Lin and C. K. Wong. Process-variation-tolerant clock skew minimization. In *1994 IEEE/ACM international conference on Computer-aided design*, pages 284–288. IEEE Computer Society Press, 1994.

[8] I.-M. Liu, T.-L. Chou, A. Aziz, and D. F. Wong. Zero-skew clock tree construction by simultaneous routing, wire sizing and buffer insertion. In *Proceedings of the international symposium on Physical design, 2000*, pages 33–38. ACM Press, 2000.

[9] T. Okamoto and J. Cong. Buffered steiner tree construction with wire sizing for interconnect layout optimization. In *Proceedings of the 1996 IEEE/ACM international conference on Computer-aided design*, pages 44–49. IEEE Computer Society Press, 1996.

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

[11] X. Tang, R. Tian, H. Xiang, and D. F. Wong. A new algorithm for routing tree construction with buffer insertion and wire sizing under obstacle constraints. In *Proceedings of the 2001 IEEE/ACM international conference on Computer-aided design*, pages 49–56. IEEE Press, 2001.

[12] J.-L. Tsai, T.-H. Chen, and C. C.-P. Chen. Epsilon-optimal minimum-delay/area zero-skew clock-tree wire-sizing in pseudo-polynomial time. In *Proceedings of the 2003 international symposium on Physical design*, pages 166–173. ACM Press, 2003.

[13] R.-S. Tsay. Exact zero skew. In *Proceedings of the 1991 IEEE international conference on Computer-aided design*, pages 336–339, 1991.

[14] L. van Ginneken. Buffer placement in distributed rc-tree networks for minimal elmore delay. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, pages 865–868, 1990.

[15] J. G. Xi and W. W.-M. Dai. Useful-skew clock routing with gate sizing for low power design. In *Proceedings of the 33rd annual conference on Design automation conference*, pages 383–388. ACM Press, 1996.