Jitter-Tolerant Clock Routing in Two-phase Synchronous Systems

Joe G. Xi* Wayne W.-M. Dai
Computer Engineering
University of California, Santa Cruz

Abstract

Due to process, manufacturing and system operating conditions in a real environment, *clock jitter* is inevitable. In the presence of jitter, zero or near-zero skew are not really safe for reliable clock operations. Appropriate skew or *useful skew* can serve as safety margin to guard against clock jitter. In two-phase clocking, the nonoverlapping interval of two-phase clocks provides an additional degree of freedom to improve either the clock tree cost or jitter-tolerance. We construct a two-phase *jitter-tolerant useful-skew tree* (JT-UST) such that the susceptibility to clock jitter and the clock tree cost is minimized. Following the Deferred-Merge Embedding (DME) framework, we use a simulated annealing approach to explore the routing topologies and embeddings. Experimental results have shown 63% to 100% reduction of jitter-prone sink pairs over previous clock routing methods while having very comparable clock tree costs.

1 Introduction

Clock routing in a synchronous system plays an important role in system performance and reliability. Algorithms to construct *zero-skew tree* (ZST) or near zero-skew tree were extensively studied[12, 2]. It was also pointed out that it is almost impossible to achieve exact zero-skew due to effects such as geometry and device parameter variations[11, 13]. Wire widening and buffer insertion methods were proposed to minimize such nonideal effects. More recent works in this area focused on reducing power dissipated in clock distribution[13, 8, 3]. With a fixed skew bound for all sink pairs, a *bounded-skew tree* (BST) can be constructed to minimize the routing cost[8, 3].

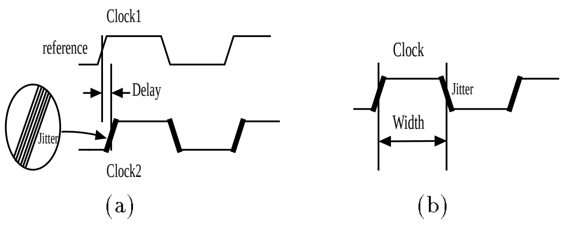

Figure 1: (a) Source to sink delay jitter; (b) Clock pulse width jitter.

*Currently with Ultima Interconnect Technology Inc, CA.

ICCAD '96
1063-6757/96 $5.00 © 1996 IEEE

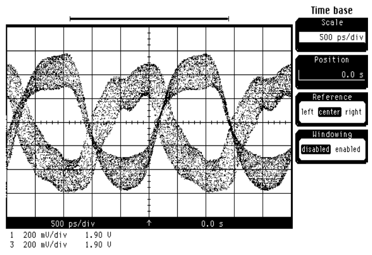

Figure 2: A clock waveform in a real environment.

In reality, there exist many uncertainties in process, manufacturing and system operating conditions. They will cause clock source to sink delays and clock pulse width to vary from their nominal values[1, 6], as illustrated in Fig. 1. Fig. 2 shows a clock waveform in a noisy environment. These random timing fluctuations of clock signals are commonly known as *clock jitter*, Some of the common causes of clock jitter are: (i) Variations of ambient temperature. (ii) Fluctuations of supply voltages. (iii) Electro-magnetic coupling to other signals. Excessive jitter can cause clock operations to fail. Moreover, jitters are often intermittent, i.e. they may not be predicted at the design stage.

In the presence of jitter, zero-skew is not necessarily safe for correct clock operation. In fact, clocking with zero-skew could sometimes be even more susceptible to clock jitter. With respect to the logic path direction, *negative* or *positive* could result in *double-clocking* or *zero-clocking* hazards, respectively[6, 14]. Consider the case when a nominal zero skew is given to a pair of clock sinks that has very small allowable positive skew but larger allowable negative skew. A potential clock jitter may cause it to be on the verge of *zero-clocking*. In this case, a reasonable negative skew between the two sinks would produce a larger *safety margin* against the hazard. The *clock skew optimization* approach tried to maximize the minimum safety margin against clock hazards[6]. Unfortunately, to realize the desired *optimal skews*, a common approach is to insert buffers as delay elements[6, 10]. But this results in increased buffer power and process variation induced skews[13].

Most previous works have concentrated on clock routing in a single-phase system with edge-triggered flip-flops. In practice, two-phase clocking scheme with master-slave level-sensitive latches is widely used. It brings several advantages to system performance and reliability, such as reduced transistor count, higher speed and testability[1]. In addition, nonoverlapping two-phase clocking creates larger safety margin away from *double-clocking*[1]. As we will discuss in this paper, the nonoverlapping interval in two-phase clocking scheme offers an additional degree of freedom to adjust safety margins against jitter.

In this paper, we seek clock routing methods to center the design away from potential hazards and achieve *jitter-tolerant* clock operations. We formulate and solve the problem of **Jitter-Tolerant Clock Routing** in nonoverlapping two-phase systems.

The rest of this paper is organized as follows. In section 2, we give the motivation and formulation of the problem. In section 3, we present our solution to the JT-UST problem. Experimental results are given in section 4.

2 Problem Formulation

Fig. 3 shows using *master-slave* double latches to build a synchronous system with two-phase clocks. Due to delays from clock source to sinks, *intra-phase skew* may exist between sinks of the same phase, e.g. between C_{01} and C_{02} or between $C_{01'}$ and $C_{02'}$. Similarly, *inter-phase skews* may also exist between sinks of different phases, such as C_{01} and C'_{01}.

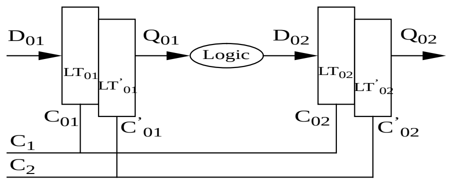

Figure 3: Two-phase nonoverlapping clocking using master-slave latches;

To ensure correct clock operation, the intra-phase skew has to satisfy:
To avoid double-clocking,

$$d_{02} - d_{01} \leq d_{nonoverlap} + MIN(d_{logic}) - d_{hold} \quad (2.1)$$

To avoid zero-clocking,

$$d_{01} - d_{02} \leq P - (d_{nonoverlap} + MAX(d_{logic}) + d_{setup}) \quad (2.2)$$

where d_{01} and d_{02} denote the delay from source to sinks C_{01} and C_{02}, respectively.

Meanwhile, the *inter-phase skew* has to be controlled such that the master-phase and slave-phase will have a nonoverlapping interval to prevent the master and the slave latches from being turned on simultaneously[1].

$$P_H + \epsilon \leq d'_{01} - d_{01} \leq P - P_H - \epsilon \quad (2.3)$$

where P_H is the clock pulse width, $\epsilon \geq 0$ is the minimum nonoverlapping interval.

Since combinational logic block delays vary from one to another, A pair of sinks may have a larger allowable *negative skew bound* (NSB) than its allowable *positive skew bound* (PSB) or vice versa. Some sink pairs are more susceptible to zero-clocking than double-clocking while others are vice versa. If a pair of sinks has a PSB barely larger than zero and it has a larger NSB, a nominal positive skew would create some safety margin away from zero-clocking, as illustrated in Fig. 4(a). Conversely, appropriate positive skew is useful to guard against double-clocking. This type of appropriate skew can thus be considered as *useful skew*. Furthermore, both NSB and PSB are functions of $d_{nonoverlap}$ which depends on the allowable inter-phase skew. If a sink pair that has small NSB but larger PSB, an increase of $d_{nonoverlap}$ would increase the safety margin against double-clocking, as shown in Figure 4(b). This gives us an additional degree of freedom to improve either the jitter-tolerance or clock tree cost.

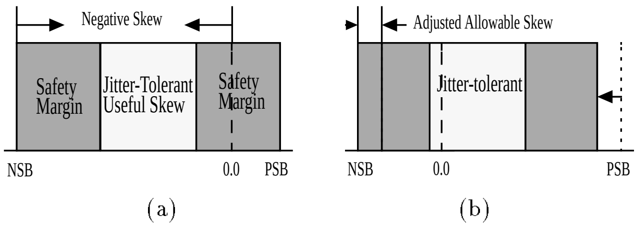

Figure 4: Safety margin against clock jitter: (a) with small allowable positive skew bound; (b) adjusting inter-phase skew to increase the negative skew bound.

For the ease of presentation, we first assume a global *clock generator* (CG) is used which generates two nonoverlapping phases at the clock source. In section 3, we will discuss how our solution can be applied in the *local clock generation scheme* where a global clock is distributed to multiple CGs and each CG generates the two phases locally[7]. Also given are the set of clock sink locations in the Manhattan plane, $S = \{s_1, s'_1, s_2, s'_2, \cdots, s_n, s'_n\}$, where s_i and s'_i correspond to the clock terminals of master and slave phases in a master-slave double latch. The *two-phase clock tree* T consists of two subtrees, T_G and T'_G, corresponding to the master- and slave-phase, respectively. Our *Jitter-Tolerant Clock Routing* problem can be stated as:

Jitter-Tolerant Clock Routing in Two-phase Systems: *Given a required clock period, P, the nominal clock pulse width, P_H, and sink locations, S, we seek a two-phase clock tree T such that the cost function $C(T)$ which measures the susceptibility to jitter and clock tree cost is minimized while each pair of sinks satisfies either the allowable inter-phase or the intra-phase skew bounds.*

3 The JT-UST Clock Routing Algorithm

3.1 Overview

Our jitter-tolerant clock tree (JT-UST) algorithm involves three tasks: (i) finding an appropriate routing

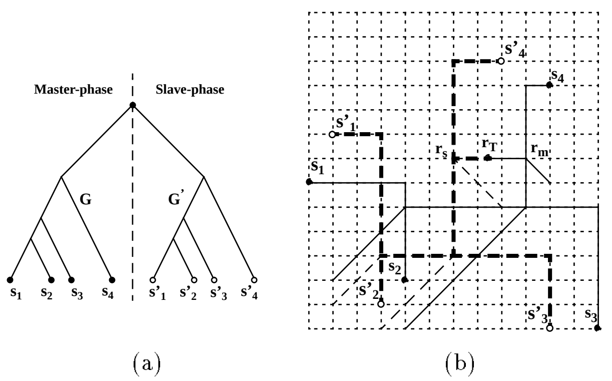

Figure 5: A two-phase clock tree example. (a) Routing topology; (b) The JT-UST constructed.

topology; (ii) embedding the internal nodes of the topology in the Manhattan plane; (iii) produce useful skews within the intra- and inter-phase skew bounds to improve jitter-tolerance. In order to carry out these tasks simultaneously, we adopt a simulated annealing approach[9].

Our JT-UST algorithm follows the framework of *Deferred Merge Embedding*(DME) approach. DME algorithms searches for feasible locations of internal nodes in the Manhattan plane[2, 8, 3]. We use a ZST constructed by the DME algorithm[2] as the initial solution. The possible locations for each node are contained in the *feasible merging segment set* (FMSS) which is determined by the allowable skew bounds. We then use the annealing approach to explore routing topologies and the placements of clock tree internal nodes. The goal is to produce the appropriate skew(useful skew) between each pair of sinks. During this process, the master- and the slave-phase trees are kept with identical topology. They are merged at the root according to the allowable inter-phase skew bounds. Elmore delay models are used for both interconnect and clock generators in evaluating the skew and constructing feasible merging segments[12, 13]. An example in Fig. 5 illustrates the general idea of our algorithm.

3.2 Skew Bounds and Jitter-Tolerance

We say two master-phase sinks, s_i and s_j are *adjacent* if there exists a combinational logic path from their corresponding latches LT_i to LT_j. Accordingly, s'_i and s'_j are also *adjacent*. The *allowable negative skew bound* (NSB) and the *allowable positive skew bound* (PSB) are the maximum value of negative and positive skew with which the clock operates correctly for a given clock frequency. Under nominal condition, the NSB and PSB between two master-phase sinks are given by: If s_i, s_j are *adjacent*, then,

$$NSB_{ij} = MIN(d_{logic}) - d_{hold} + P - 2P_H - \epsilon, \quad (3.4)$$

$$PSB_{ij} = P - MAX(d_{logic}) - d_{setup} - \epsilon \quad (3.5)$$

If s_i, s_j are not *adjacent*, then,

$$NSB_{ij} = \infty, \qquad PSB_{ij} = \infty \quad (3.6)$$

The NSB and PSB between slave-phase sinks are similarly defined. Under nominal condition, the *allowable inter-phase skew upper bound* (USB) and *allowable inter-phase skew lower bound* (LSB) are given by:

$$USB_{ii'} = P - P_H - \epsilon, \qquad LSB_{ii'} = P_H + \epsilon \quad (3.7)$$

Assume the maximum delay jitter from source to sink is $\pm M_1/2$ and the maximum pulse width jitter is $\pm M_2$, we may make the following observations:

Observation 1: *If we adjust the NSB and PSB by: $NSB^J_{ij} = NSB_{ij} - M_1 - M_2$, $PSB^J_{ij} = PSB_{ij} - M_1 - M_2$ and the LSB and USB by: $LSB^J_{ii'} = LSB_{ii'} + M_2$, $USB^J_{ii'} = USB_{ii'} - M_2$, and if intra- and inter-phase skews satisfy the adjusted bounds under nominal condition, correct clock operations may still be achieved even with additional intra-phase skew of M_1 and pulse width jitter of M_2. Therefore, clock operations are jitter-tolerant.*

Observation 2: *The largest safety margin for intra-phase skew is $1/2(PSB_{ij} + NSB_{ij})$ when $d_{ij} = d_i - d_j = 1/2(PSB_{ij} - NSB_{ij})$ is the most jitter-tolerant intra-phase skew and the largest safety margin for inter-phase skew is $1/2(P - 2P_H - 2\epsilon)$ when $d_{ii'} = d'_i - d_i = 1/2P$ is the most jitter-tolerant inter-phase skew.*

Measuring the safety margin(or jitter-tolerance) is equivalent to measuring how close a nominal skew d_{ij} or $d_{ii'}$ is to its most jitter-tolerant value. We define this closeness as the *susceptibility to jitter*, $\mathcal{J}$[1]:

$$\mathcal{J}_{ij} = |d_{ij} - 1/2(PSB_{ij} - NSB_{ij})|, \qquad \mathcal{J}_{ii'} = |d_{ii'} - 1/2P| \quad (3.8)$$

We may define the cost function of a clock tree, $C(T)$ to measure both the clock tree cost and jitter-tolerance:

$$C(T) = \lambda L(T) + \gamma\Big(\sum_{s_i, s_j \in S_A} \omega_{ij}\mathcal{J}_{ij} + \sum_{s_i, s'_i \in S} \omega_{ii'}\mathcal{J}_{ii'}\Big) \quad (3.9)$$

where λ and γ are coefficients to appropriately align the clock tree cost and jitter-tolerance. ω_{ij} and $\omega_{ii'}$ are weights for each pair of sinks.

3.3 Feasible Placements of Clock Tree Nodes

Associated with each node in the master-phase tree, $v \in G$, we define $NSB(v)$ and $PSB(v)$ as the maximum allowable delay difference from v to its two children, a and b.

(I) If the two children nodes of v are sinks, i.e. s_i, s_j, then $NSB(v) = NSB_{ij}$ and $PSB(v) = PSB_{ij}$.

(II) If one or more of the children nodes of v are not sinks, i.e. a and b with subtrees, T_a and T_b, then,

$$NSB(v) = \min(d_i(a) - d_j(b) + NSB_{ij}, d_l(a) - d_k(b) + PSB_{kl}) \quad (3.10)$$

[1]In other words, the smaller $\mathcal{J}$ for a given skew, the larger safety margin it allows against clock jitter.

$$PSB(v) = \min(d_j(a) - d_i(b) + PSB_{ij}, d_k(b) - d_l(a) + NSB_{kl}) \quad (3.11)$$

for all sink pairs, $s_i, s_l \in T_a, s_j, s_k \in T_b$. $d_i(a), d_l(a)$ and $d_j(b), d_k(b)$ denote the delays from a to s_i, s_l and b to s_j, s_k, respectively. A feasible placement of v has to satisfy $NSB(v)$ and $PSB(v)$ in order to satisfy the skew bounds between sinks rooted at v. For each node in the slave-phase tree, $v' \in G'$, $NSB(v')$ and $PSB(v')$ are similarly defined with $NSB_{i'j'}$ and $PSB_{i'j'}$.

We define $USB(r_T)$ and $LSB(r_T)$ as the maximum allowable delay difference from r_T to its two children, r_m and r_s, which are the root nodes of the master-phase and slave-phase trees, respectively.

$$USB(r_T) = \max(d_i(r_m) - d_{i'}(r_s) + USB_{ii'}), \quad (3.12)$$

$$LSB(r_T) = \min(d_i(r_m) - d_{i'}(r_s) + LSB_{ii'}) \quad (3.13)$$

for all sink pairs, $s_i \in T_G, s'_i \in T_{G'}$. $d_i(r_m)$ and $d_i(r_s)$ denote the delays from r_m to s_i and r_s to $s_{i'}$, respectively.

We see that the routing topology and the embedding of the clock tree will determine how large our feasible solution space is and consequently how much jitter-tolerance resulted. To explore various topologies and feasible placements of internal nodes, we adopt a *Subtree Swapping* (SSP) and a *Merging Segment Perturbation* (MSP) procedures.

A *Subtree Swapping* (SSP) is an operation that swaps two subtrees rooted at two non-sibling nodes $v_i, v_j \in G$, denoted as $SSP(v_i, v_j)$. For each $SSP(v_i, v_j)$, an $SSP(v'_i, v'_j)$ is also performed for the corresponding $v'_i, v'_j \in G'$ to keep G and G' identical. After each $SSP(v_i, v_j)$, the feasible placements for ancestor nodes on the path from v_i, v_j to r_m including the root of T, r_T are updated. So are the ancestor nodes of v'_i, v'_j. Updating of feasible placements is the same as in an MSP. The *Merging Segment Perturbation* (MSP) procedure can be referred to in [14].

3.4 Extension to Local Clock Generation Scheme

In the above, we described the algorithm to route two-phase clock trees with a single clock generator at the source. This scheme is shown in Fig. 6(a). Fig. 6(b) shows an alternative way of distributing two-phase clocks, the *local clock generation scheme*. A system clock is distributed globally to multiple CGs. Each CG is used to generate two phases in a local cluster. This way, we may reduce the global wiring at the expense of using more CGs. Here we briefly describe how our JT-UST algorithm can be extended to this scheme.

The local clock generation scheme can be considered as having a two-level clock tree. The *global* clock tree is the system clock tree with leaf nodes corresponding to the local CGs. The *local* clock trees are the two-phase clock trees each of which is rooted at the output of a CG with leaf nodes corresponding to the sinks within a cluster. Since each CG is usually designed

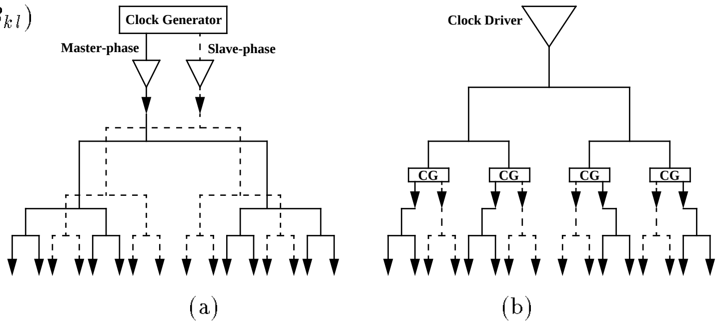

Figure 6: (a) Global clock generation; (b) Local clock generation.

and characterized with specific load requirement, the size of each cluster should be based on the total capacitive load. We adopt a *nearest-neighbor merging* approach[5]. Each cluster is formed by grouping the nearest neighbors until the total load of a cluster reaches the load limit of a CG. Within each cluster, the local clock tree can be considered as a two-phase tree in which the intra- and inter-phase allowable skew bounds described earlier can be used. The global clock tree is a single-phase tree in which only the allowable intra-phase skew bounds have to be satisfied.

4 Experimental Results

Our algorithm has been implemented in C in a Sun Sparcstation10 environment and has been tested on two industry circuits and three ISCAS89 benchmark circuits as shown in Table 1. The ISCAS89 benchmark circuits were modified to a $0.65\mu m$ CMOS standard-cell library[4]. To evaluate the overall jitter-tolerance, we would like to see how many sinks are jitter-prone, i.e. causing clock operations to fail in the presence of jitter. In general, it is unlikely that the same amount of jitter will occur to every pair of sinks. In our cost function, larger weights are given to the sink pairs that are farther apart, i.e. the first ancestor node common to the two sinks is closer to the root. Therefore, we rank the nodes in a clock tree into 3 levels according to their heights. The sinks belong to different subtrees at each level are given a delay jitter of $10\%P, 5\%P, 2.5\%P$, a pulse width jitter of $15\%P_H, 10\%P_H, 5\%P_H$, correspondingly. We also run the DME based ZST and BST clock routing algorithms[2, 3, 5] to compare with our JT-UST solution. In the BST algorithm, the smallest allowable skew bound(both negative and positive) of all sink pairs have to be chosen as the fixed intra-phase skew bound. Table 2 gives the number of sink pairs that exceeded the allowable skew bounds with given jitter. JT-UST reduces at least 63% of the number jitter-prone sink pairs over either ZST or BST approaches. Table 3 compares the routing results of ZST, BST and JT-UST. The JT-UST approach achieves savings of wire length over ZST in three cases and only increases marginally in the other two cases.

Circuits	Delay Jitter			Pulse Width Jitter			Delay & Pulse Width Jitter			Reduction	
	ZST	BST	JT-UST	ZST	BST	JT-UST	ZST	BST	JT-UST	$\frac{JT-UST}{ZST}$	$\frac{JT-UST}{BST}$
Circuit1	11	8	0	1	1	0	11	8	0	100%	100%
Circuit2	18	11	2	4	3	1	21	15	4	81%	73%
s1423	3	3	0	0	0	0	3	4	0	100%	100%
s5378	23	20	5	7	4	2	34	31	8	76%	74%
s15850	32	34	12	10	9	2	43	45	16	63%	64%

Table 2: The number of jitter-prone sinks, i.e. which caused either double-clocking or zero-clocking.

Circuits	Freq (Mhz)	latches #	P_H (ns)	CG #
Circuit1	200	106	1.7	1
Circuit2	100	391	3.3	8
s1423	33	74	10.0	1
s5378	100	179	3.3	3
s15850	100	597	3.3	16

Table 1: Circuits tested.

Circuits	ZST	BST (Skew-bound)	JT-UST
Circuit1	8082	7893 (0.05 ns)	7851
Circuit2	23369	22528 (0.12 ns)	22912
s1423	16507	14283 (0.9 ns)	16902
s5278	25756	24552 (0.1 ns)	24860
s15850	43897	42010 (0.08 ns)	43988

Table 3: Comparison of wire length(in μm) of clock trees on tested circuits.

5 Conclusions

In a real environment where clock jitters are unavoidable, zero-skew is not really safe for reliable clock operations. We proposed *jitter-tolerant clock routing* to produce safety margin guarding against clock jitter. In two-phase clocking, the nonoverlapping interval can be used to add additional solution space to either improve jitter-tolerance or clock tree cost.

6 Acknowledgement

We wish to thank C. W. Albert Tsao and Prof. Andrew Kahng of UCLA for providing us with the program of DME algorithms.

References

[1] H. Bakoglu. *Circuits, Interconnections, and Packaging for VLSI.* Addison-Wesley Publishing Company, 1987.

[2] T.H. Chao, Y.C. Hsu, J.M.Ho, K. D. Boese, and A. B. Kahng. Zero skew clock net routing. *IEEE Transactions on Circuits and Systems*, 39(11):799–814, November 1992.

[3] J. Cong, A.B. Kahng, C.K. Koh, and C-W. A. Tsao. Bounded-skew clock and steiner routing under elmore delay. In *To appear in IEEE Intl. Conf. on Computer Aided Design*, 1995.

[4] National Semiconductor Corp. *cs65 CMOS Standard Cell Library Data Book.* National Semiconductor Corp, 1993.

[5] M. Edahiro. A clustering-based optimization algorithm in zero-skew routings. In *Proc. of 30th ACM/IEEE Design Automation Conference*, pages 612–616, 1993.

[6] J. P. Fishburn. Clock skew optimization. *IEEE Transactions on Computers*, 39(7):945–951, 1990.

[7] S. Ganguly and S. Hojat. Clock distribution design and verification for powerpc microprocessors. In *IEEE Intl. Conf. on Computer Aided Design*, pages 58–61, Nov. 1995.

[8] D. J.-H. Huang, A. B. Kahng, and C-W. A. Tsao. On the bounded-skew clock and steiner routing problems. In *Proc. of 32nd Design Automation Conf.*, pages 508–513, 1995.

[9] S. Kirkpatrick, Jr. C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):458–463, May 1983.

[10] J. L. Neves and E. G. Friedman. Design methodology for synthesizing clock distribution networks exploiting non-zero localized clock skew. *Manuscript*, 1994.

[11] S. Pullela, N. Menezes, J. Omar, and L. T. Pillage. Skew and delay optimization for reliable buffered clock trees. In *IEEE Intl. Conf. on Computer Aided Design*, pages 556–562, 1993.

[12] R-S. Tsay. An exact zero-skew clock routing algorithm. *IEEE Trans. on Computer-Aided Design*, 12(3):242–249, 1993.

[13] Joe G. Xi and Wayne W.M. Dai. Buffer insertion and sizing under process variations for low power clock distribution. In *Proc. of 32nd Design Automation Conf.*, June 1995.

[14] Joe G. Xi and Wayne W.M. Dai. Useful-skew clock routing with gate sizing for low power designs. In *33rd Design Automation Conf.*, June 1996.