On Clock Routing For General Cell Layouts

Jason Cong, Andrew Kahng, Gabriel Robins

UCLA Department of Computer Science, Los Angeles, CA 90024-1596

Abstract

This paper addresses the minimum-skew clock routing problem in general cell designs. We present a bottom-up construction method for clock distribution trees based on a generalized matching computation in channel intersection graphs. The clock routing trees produced by our method attain almost zero skew with only modest wirelength penalty. Experimental results show that clock skew of our routing tree is no more than 2% of that of a minimum spanning tree, while on average the cost of the tree is only 50% more than that of a minimum spanning tree.

1 Introduction

In synchronous high-performance VLSI systems, limitations on circuit speed are determined by two factors: the delay on the longest path through combinational logic and the maximum clock skew among the synchronizing components. Due to advances in VLSI fabrication technology, delay through combinational logic has decreased considerably, and thus the clock skew induced by non-symmetric clock distribution has become a more significant limitation on circuit performance.

A number of researchers have recently studied clock skew minimization. H-tree constructions have been used extensively for clock routing in regular systolic arrays [1] [4] [5] [13]. Although the H-tree structure can significantly reduce clock skew [4] [13], it is applicable primarily when the synchronizing components are identical in size and are placed in a regular array.

Ramanathan and Shin [10] proposed a clock distribution scheme for building block design where all blocks are organized in a hierarchical manner. They assume that a clock entry point is given at each level of the hierarchy and, moreover, that the number of blocks at each level is small since an exhaustive search algorithm is used to enumerate all possible routes.

Jackson, Srinivasan and Kuh [6] presented a clock routing scheme for circuits with many small cells. Their algorithm recursively partitions a circuit into two equal parts, and then connects the center of mass of the whole circuit to the centers of mass of the two sub-circuits. Their algorithm is efficient and easy to implement. However, although it was shown that the maximum difference in path length from the root to different synchronizing components is bounded by $O(\frac{1}{\sqrt{n}})$ on average, one may construct examples for which wirelengths between clock source and clock pins in their solution may vary by as much as the entire chip diameter.

We recently presented a clock routing scheme for cell-based designs that achieves very small clock skews while still using a reasonable amount of wire [7]. This routing solution constructs a binary routing tree via recursive geometric matching. We showed that on average, the total wire length of such a path-balanced tree is within a constant factor of the wirelength for an optimal Steiner tree; in the worst case, the tree cost is bounded by $O(\sqrt{n})$ when the n leaves are arbitrarily distributed in the unit square. Empirical results on both random examples and industrial benchmarks were very promising: near-zero average clock skew was achieved using similar or even shorter total wire length than the method of [6].

In this paper, we study the high-performance clock routing problem in general cell designs. In general cell design style, a circuit is partitioned into a set of general cells (also referred to as blocks). Routing is carried out in the channels between blocks and routing over blocks is prohibited. Feasible routing regions are represented by the channel intersection graph [3]. Hence, cell-based routing methods do not apply since the routing cost between two points can no longer be approximated by the Manhattan distance between the points. Moreover, blocks are usually of different sizes and are not placed in a regular array structure; thus the H-tree clock routing scheme cannot be applied. In this paper, we extend the matching-based approach of [7] to clock routing for general cell designs. Our algorithm minimizes skew using a bottom-up recursive matching scheme to construct a clock tree that is balanced with respect to root-leaf pathlengths in the tree. Experimental results are very favorable: our algorithm achieves near-zero pathlength skew with only a modest penalty in wiring cost. SPICE simulation results show that the skew of our clock tree is much less than that of a minimum spanning tree.

The remainder of this paper is organized as follows. Section 2 defines a number of basic concepts and gives a precise formulation of our skew minimization problem. In Section 3, we present the clock routing algorithm in detail. Experimental results of our algorithm are presented in Section 4.

91TH0379-8/91/0000-P14-5.1$01.00 © 1991 IEEE

2 Preliminaries

A synchronous VLSI circuit consists of two types of elements: synchronizing elements and combinational logic gates. The synchronizing elements are connected to one or several system-wide clock signals. Every closed path in a synchronous circuit contains at least one synchronizing element. The speed of a synchronizing circuit is mainly determined by the clock periods. It is well known [2] [6] that the clock period C_P of each clock signal net satisfies the inequality

$$C_P \geq t_d + t_{skew} + t_{su} + t_{ds}$$

where t_d is the delay on the longest path through combinational logic, t_{skew} is the clock skew, t_{su} is the set up time of the synchronizing elements (assuming that the synchronizing elements are edge triggered), and t_{ds} is the propagation delay within the synchronizing elements. As VLSI feature sizes become smaller, the terms t_d, t_{su}, and t_{ds} all decrease considerably. Therefore clock skew becomes a more dominant factor in determining circuit performance.

Given a routing solution for a clock signal net, the clock skew is defined to be the maximum difference among the delays from the *clock entry point* (CEP) to synchronizing elements in the net. The delay from the CEP to any synchronizing element depends on the wire length from the CEP to the synchronizing element, RC constants of wire segments in the routing, and the topology of the solution (assuming that the load capacitance at the gate of each synchronizing element is the same). Usually, the clock routing may be described as a RC tree [11], and we commonly use the first-order moment of the impulse response (also called Elmore's delay) to approximate delay in an RC tree. The formulas derived in [11] give both upper and lower bounds on delay in an RC tree, thus yielding a more accurate approximation.

However, although both the formula for Elmore's delay and those in [11] are very useful for simulation or timing verification, they involve sums of quadratic terms and are difficult to compute and optimize during the layout design process. Thus, a linear RC model, where the delay is proportional to the wirelength between CEP and the synchronizing element, is often used to derive a simpler approximation for circuit delay (e.g., [10] [9]). In this paper, we also use wire length as a simple approximation of the delay in a routing solution. The clock skew is defined to be the maximum difference in pathlength from the CEP to synchronizing elements in the clock signal net. The SPICE simulation results in Section 4 confirm that minimizing pathlength difference leads to clock trees with very small skews.

In general cell designs, a clock routing solution is represented by a rooted (Steiner) tree in the channel intersection graph whose root is the CEP and whose leaves are the clock pins of synchronizing elements in the clock signal net. Each internal node of the clock routing tree is a channel intersection point. Note that each edge lies completely in a channel. The length, or cost, of an edge in the tree is the span of the edge in the channel. The tree cost is the sum of all edge costs in the tree.

Definition: The *pathlength skew* of a tree is defined to be the maximum difference of the pathlengths in the tree from the root to any two leaves.

A tree is called a *perfect pathlength balanced tree* if its pathlength skew is zero. Note that it is not difficult to construct a perfect pathlength balanced tree if we are allowed to waste an arbitrary amount of wire. However, such a construction will lead to a routing tree with very high cost, implying a large RC constant that may distort the clock signal. Thus, we wish to construct a clock routing tree whose pathlength skew is as small as possible, while the total tree cost is not too large. With this in mind, we formulate the clock routing problem as follows:

The Path Balanced Tree (PBT) Problem: Given a clock net N and real numbers B and S, find a clock routing tree such the pathlength skew of the tree is no more than S and the tree cost is bounded by B.

It is not difficult to show that the PBT problem is NP-complete [7]. The objective of this paper is to present an efficient heuristic algorithm for the PBT problem in general cell designs.

Given a graph G with a cost function on the edges, we let $minpath_G(x, y)$ denote the minimum cost path between nodes x and y, and use $dist_G(x, y)$ to denote the cost of $minpath_G(x, y)$. We define the concept of a generalized matching as follows:

Definition: Given a graph $G = (V, E)$ with a cost function on the edges, a *generalized matching* M in G is a set of shortest paths connecting m mutually disjoint node pairs, i.e., $M = \{minpath_G(x_1, y_1), minpath_G(x_2, y_2), \ldots, minpath_G(x_m, y_m)\}$, where the x_i's and y_i's are all distinct.

A generalized matching on a set of nodes N in G is *complete* if $m = \lfloor \frac{|N|}{2} \rfloor$. The *cost* of a generalized matching M is the sum of the costs of the shortest paths in the matching, $cost(M) = \sum_{i=1}^{m} dist_G(x_i, y_i)$. An *optimal* complete generalized matching on $N \subseteq V$ is one with least cost. We can show the following properties of optimal complete generalized matchings:

Lemma 1: Each edge of G belongs to at most one shortest path in an optimal complete generalized matching on $N \subseteq V$ in G.

Proof: Let M be an optimal complete generalized matching on N. Suppose that edge e appears in both $minpath_G(x_i, y_i)$ and $minpath_G(x_j, y_j)$. It is not difficult to show that

$$dist_G(x_i, x_j) + dist_G(y_i, y_j) \leq$$
$$dist_G(x_i, y_i) + dist_G(x_j, y_j) - 2 \cdot cost(e)$$

Therefore, we would obtain a complete generalized matching on N with smaller cost if we replace $minpath_G(x_i, y_i)$ and $minpath_G(x_j, y_j)$ by

$minpath_G(x_i, x_j)$ and $minpath_G(y_i, y_j)$ in M, which is a contradiction. □

Henceforth, on we will assume that the circuit layout is in a LxL grid. There are n blocks in the design and and there are k terminals in the clock signal net. G is the underlying channel intersection graph together with the k clock terminals.

Lemma 2: The routing cost between any two clock terminals in G is no more than $2L$.

Proof: Let x and y be two clock terminals in G. Let P_1 be a monotone (staircase) path passing through x and connecting two opposite corners w and w' of the layout. Clearly, $cost(P_1) \leq 2L$. Similarly, let P_2 be a monotone path passing through y and connecting w and w'. Then, $cost(P_1) + cost(P_2) \leq 4L$. It follows that at least one of w or w' can be reached from *both* x and y with cost at most $2L$. Thus, the shortest path between x and y has cost no more than $2L$. □

It is clear from this lemma that an optimal complete generalized matching on the clock terminals in G has cost no more than $2L \cdot \lfloor \frac{k}{2} \rfloor \leq kL$.

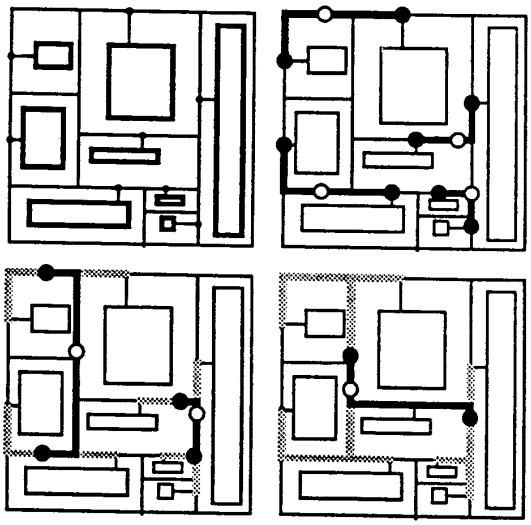

Figure 1: Example of the algorithm running on a random placement of modules with an 8-terminal net. Solid dots are roots of subtrees in the previous level; hollow dots are roots of new subtrees computed at the current level. At each level an optimal complete generalized matching is computed on the solid points. For clarity, only the newly added wires are highlighted at each level.

3 A Clock Routing Algorithm

We construct a clock tree on the clock terminals in G by computing a sequence of generalized matchings. We begin with a forest of k isolated clock terminals (for convenience, assume $k = 2^l$), each of which is a degenerate tree with CEP being the node itself. The optimal complete generalized matching on these k terminals yields $\frac{k}{2}$ paths, each of which defines a subtree. The optimal CEP into each subtree of two nodes is the midpoint of the corresponding path, so that the clock signal will have zero skew between the path endpoints. In general, at each level, we compute an optimal generalized matching on the set of CEPs (i.e., roots) of all subtrees in the current forest and merge each pair of subtrees into a larger subtree. We choose the root (CEP) of resulting tree to be the *balance point* on the path connecting the two subtrees so that the path-length skew in the resulting tree is minimized (Figure 1).

Notice that at each level of the recursion, we only have to match half as many nodes as in the previous level. Thus, in $l = \lceil \log k \rceil$ matching iterations, we obtain a complete clock tree topology. If k is not a power of 2, then there is an odd number $2j+1$ of nodes to match at some level. In this case, we compute an optimal complete generalized matching on $2j$ nodes, then match $j+1$ nodes at the next level. Figure 2 gives a formal description of the algorithm.

```
T = ∅
P = N
while |P| > 1
    M = opt complete generalized matching on P
    P' = ∅
    for {p1, p2} ∈ M do
        T1 = subtree of T rooted at p1
        T2 = subtree of T rooted at p2
        p = balance point on minpath_G(p1, p2)
              minimizing the skew of the tree
              T1 ∪ T2 ∪ minpath_G(p1, p2)
        P' = P' ∪ {p}
        T = T ∪ {{p, p1}, {p, p2}}
    P = P' plus an unmatched node if |P| is odd
CEP = Root of T = single remaining point in P
```

Figure 2: The recursive matching-based clock tree algorithm for a set of terminals N in general cell design.

The worst-case clock tree cost produced by the algorithm can be bounded as follows:

Theorem: Given n blocks in the LxL grid and k terminals of a clock signal net, the cost of the clock tree created by our algorithm is at most $2kL$.

Proof: According to Lemma 2, the cost of an generalized matching on k terminals is bounded by kL. After each iteration, the number of nodes to be matched is reduced by half. Therefore, the total clock tree cost is bounded by $kL + \frac{kL}{2} + \frac{kL}{4} + \frac{kL}{8} + \ldots \leq 2kL$. □

In order to compute an optimal generalized matching on a set of nodes N in G, we construct a weighted complete graph G' on N such that $weight(x, y) = dist_G(x, y)$ for each pair of nodes x and y in N. Then, we may apply the $O(n^3)$ algorithm for computing an optimal complete matching in general graphs [8]. This leads to an $O(k^3 log k)$ time routing algorithm for a clock net of size k. However, this complexity will result in long runtimes for large problem instances. In order to achieve an efficient implementation, we

use the greedy matching heuristic [12], an approach that proved successful in [7]. The complete generalized matching solution based on the greedy heuristic is improved by removing overlapping edges of shortest paths, as in the proof of Lemma 1, so that no edge is used in more than one shortest path.

4 Experimental Results

Our algorithm was implemented in ANSI C on Sun SPARC workstations. Code is available from the authors. We have tested our algorithm on two sets of test cases. One set of examples contains clock nets of size 4, 8, and 16 on 16 blocks, and the other set contains clock nets of size 4, 8, and 16 on 32 blocks. For each net size, 100 instances were generated randomly; we compare the skew and cost of the routing trees produced by our algorithm with corresponding minimum spanning tree (MST) values. Results are shown in Tables 1 and 2. Clearly, the skew of our clock tree is very close to zero. In no case it is more than 2% of that of a MST. The increase of total wirelength of our routing tree varies from 24% to 77% when compared with a MST.

We also ran SPICE simulations on a number of examples (using MOSIS 2.0μ CMOS technology and 0.3pF gate loading capacitance). The actual skew of our clock tree is consistently much smaller than that of a minimum spanning tree. For a typical 16-pin clock net in a 16-block design, the skews of our clock tree and the MST are 18ps and 69ps, respectively. The wirelengths of the two trees are 1775 and 1043, respectively.

For the routing tree produced by our algorithm, we may have overlapping edges in a channel because matching paths at different levels may use the same channel. However, by Lemma 1, no channel will appear in more than one path in a single matching. Therefore, there are at most $\lceil logk \rceil$ overlapping edges in each channel. The last column in Table 2 shows the average edge density in channels, computed as the average of non-zero local column densities over all columns in all channels.

5 Concluding Remarks

We presented a bottom-up approach for constructing clock routing trees in general cell designs. Skew minimization is achieved by carefully balancing the pathlengths from the root to all leaves. We are also investigating the possibility of varying driver sizes and using short poly lines to further minimize the skew. These techniques have the potential of completely eliminating clock skew with very little or no increase in routing cost.

References

[1] H. Bakoglu, J. T. Walker and J. D. Meindl, "A Symmetric Clock-Distribution Tree and Optimized High- Speed Interconnections for Reduced Clock Skew in ULSI and WSI Circuits", *Proc. IEEE ICCD*, Oct. 1986, pp. 118-122.

[2] H. Bakoglu, *Circuits, Interconnections and Packaging for VLSI* , Addison-Wesley, 1990.

[3] W. M. Dai, T. Asano and E. S. Kuh, "Routing Region Definition and Ordering Scheme for Building-Block Layout", *IEEE Trans. on CAD*, Vol. CAD-4, July 1985, pp. 189-197.

[4] S. Dhar, M. A. Franklin and D. F. Wann, "Reduction of Clock Delays in VLSI Structures", *Proc. IEEE ICCD*, Oct. 1984, pp. 778-783.

[5] A. L. Fisher and H. T. Kung, "Synchronizing Large Systolic Arrays", *Proc. SPIE* 341, May 1982, pp. 44-52.

[6] M. A. B. Jackson, A. Srinivasan and E. S. Kuh, "Clock Routing for High-Performance ICs", *Proc. ACM/IEEE DAC*, June 1990, pp. 573-579.

[7] A. Kahng, J. Cong and G. Robins, "High-Performance Clock Routing Based on Recursive Geometric Matching", *Proc. ACM/IEEE DAC*, June 1991, to appear.

[8] E. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt Rinehart and Winston, New York, 1976.

[9] I. Lin and H. C. Du, "Performance-Driven Constructive Placement", *Proc. ACM/IEEE DAC*, June 1990, pp. 103-105.

[10] P. Ramanathan and K. G. Shin, "A Clock Distribution Scheme for Non-Symmetric VLSI Circuits", *Proc. IEEE ICCAD*, November 1989, pp. 398-401.

[11] J. Rubinstein, P. Penfield and M. A. Horowitz, "Signal Delay in RC Tree Networks", *IEEE Trans. on CAD* CAD-2(3) (1983), pp. 202-211.

[12] T. L. Snyder and J. M. Steele, "Worst-Case Greedy Matchings in the Unit d-Cube", *Networks* 20 (1990), pp. 779-800.

[13] D. F. Wann and M. A. Franklin, "Asynchronous and Clocked Control Structure for VLSI Based Interconnection Networks", *IEEE Trans. on Computers* 21(3) (1983), pp. 284-293.

# of modules	net size	skew		cost	
		MST	ours	MST	ours
16	4	511.0	0.8	1537	1921
16	8	794.9	12.9	2328	3478
16	16	1101.5	22.1	3332	5873
32	4	445.0	0.4	1401	1729
32	8	804.4	4.4	2261	3407
32	16	1136.9	12.0	3357	5847

Table 1: Average clock tree costs and pathlength skews, in grid units, of our method and those of the MST, respectively. For each row, 100 random instances were generated and routed on a 1000 by 1000 grid.

# of modules	net size	clock skew	tree cost	edge density in channels
16	4	0.00	1.26	1.24
16	8	0.02	1.49	1.40
16	16	0.02	1.77	1.63
32	4	0.01	1.24	1.21
32	8	0.01	1.52	1.36
32	16	0.01	1.74	1.48

Table 2: Average clock tree costs and pathlength skews of our method, normalized (per instance) to corresponding MST values. For each row, 100 random instances were generated and routed on a 1000 by 1000 grid.