ASSOCIATIVE SKEW CLOCK ROUTING FOR DIFFICULT INSTANCES

A Thesis

by

MIN-SEOK KIM

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2006

Major Subject: Electrical Engineering

ASSOCIATIVE SKEW CLOCK ROUTING FOR DIFFICULT INSTANCES

A Thesis

by

MIN-SEOK KIM

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Jiang Hu
Committee Members,	Weiping Shi
	Eun J. Kim
Head of Department,	Costas N. Georghiades

May 2006

Major Subject: Electrical Engineering

ABSTRACT

Associative Skew Clock Routing for Difficult Instances. (May 2006)

Min-seok Kim, B.S., Texas A&M University

Chair of Advisory Committee: Dr. Jiang Hu

This thesis studies the associative skew clock routing problem, which seeks a clock routing tree such that zero skew is preserved only within identified groups of sinks. Although the number of constraints is reduced, the problem becomes more difficult to solve due to the enlarged solution space. Perhaps, the only previous study used a very primitive delay model which could not handle difficult instances when sink groups are intermingled. We reuse existing techniques to solve this problem including difficult instances based on an improved delay model. Experimental results show that our algorithm can reduce the total clock routing wirelength by 9%–15% compared to greedy-DME, which is one of the best zero skew routing algorithms.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

In the design of high performance VLSI systems, the clock distribution network design is one of the most critical tasks. The wirelength of clock network should be minimized in order to reduce system power requirements, power/ground supply noise, process variations and thermal problems. A clock routing instance consists of a set of sink register locations $\mathbf{S} = \{S_1, S_2, \ldots, S_n\} \subset \mathbf{R}^2$ in the Manhattan plane and optionally a connection topology G rooted at source s_0, with n leaves corresponding to sinks in **S**. The clock routing problem constructs a clock tree T(**S**) with topology G to minimize total wirelength subject to certain skew constraints. For a given clock tree T(**S**), let $t_d(s_0, s_i)$ denote the signal propagation time on the unique path from source s_0 to sink s_i. The skew of T(**S**) is the maximum value of $|t_d(s_0, s_i) - t_d(s_0, s_j)|$ over all sink pairs $s_i, s_j \in$ **S**. If T(S) has zero skew then it is called a zero skew clock tree (ZST). Skew constraints ensure that circuits operate properly at desired frequency. In general, skew constraint may take one of the following forms:

- *Zero skew* [1–3]. This requires that clock signal arrival times are the same for all sinks. There is Deferred-Merge Embedding (**DME**) algorithm that uses zero skew to construct the clock routing tree. See Figure 1(a). Zero skew constraint is very popular in industry due to its simplicity.

- *Bounded skew*. In this case, a skew does not have to be zero and can be any value within a bound. The bound can be either a global one for all sinks [4] or a local one for each pair of clock sinks [5]. The relaxed skew constraints normally lead to smaller clock network size compared to zero skew routing. See Figure 1(b).

This thesis follows the style of *IEEE Transactions on Computer-Aided Design.*

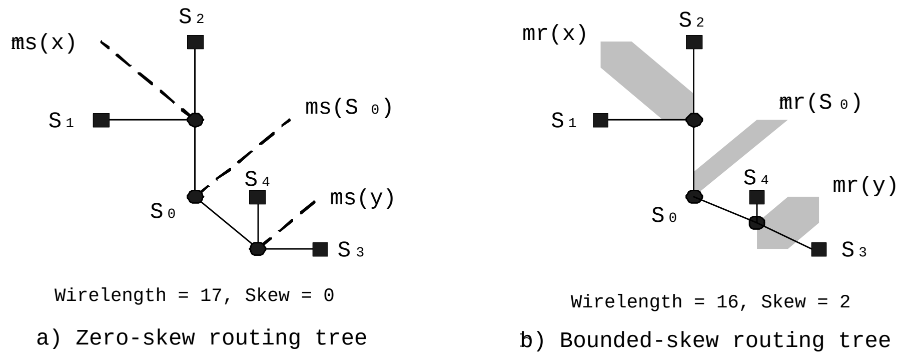
ms(x)
S2
S1
ms(S0)
S4
ms(y)
S0
S3
Wirelength = 17, Skew = 0
a) Zero-skew routing tree
mr(x)
S2
S1
mr(S0)
S4
mr(y)
S0
S3
Wirelength = 16, Skew = 2
b) Bounded-skew routing tree

Fig. 1. Comparison of **DME** zero-skew routing in (a) and **BST/DME** bounded-skew routing in (b). The relaxed skew bound of bounded-skew routing gives less total wirelength than that of zero-skew routing.

- *Prescribed skew* [6,7]. The skew for each pair of sinks is required to satisfy a usually non-zero target. This is for the purpose of improving circuit operating frequency [8, 9], reducing power supply noise [10] or improving tolerance to variations [11].

Another form of skew constraint, associative skew, has been rarely mentioned in literature before. In associative skew clock routing [12], the classic “zero-skew clock tree” considered over-constrained – in that there are constraints between all pairs of sinks. However, skew constraints actually exist only between pairs of sequentially adjacent registers. Indeed, “clock skew between non-sequentially connected registers, from an analysis viewpoint, has no effect on the performance and reliability of the synchronous system and is essentially meaningless” [13]. Therefore, in an associative skew clock routing problem [12], they divide sinks into several groups. Skew constraints are required only within each group and do not exist between different groups. This problem is more difficult to solve than conventional zero/prescribed skew routings because the reduction of constraints results in increased solution space to be searched.

To the best of our knowledge, [12] is the only previous work attempting to solve this problem. Although several heuristic solutions were proposed in [12], all of them are based on a very primitive delay model which equalizes clock signal delay with geometrical pathlength. In practice, the pathlength (linear) delay model does not provide reliable control of actual delay skew. Moreover, there are effective only when the sink groups are geometrically separated from each other. In difficult but common instances where the sink groups are intermingled with each other, the algorithm developed in [12] perform worse than a simple extension to greedy-DME [2] which is one of the best zero skew routing algorithms.

Why is there so few work on this problem? Perhaps people have a simple solution in practice. That is, enforce arbitrary or empirical skew constraints between each pair of groups so that this problem becomes a conventional clock routing problem. For example, one might require the skew among all groups to be zero if intra-group skew constraint is zero for all groups. In fact, this is the extension to greedy-DME [2] made in [12] for comparing their algorithms. However, it is not obvious how to find the best inter-group constraints for minimizing clock routing wirelength. Our experiments show that an elaborated method can reduce wirelength by about 15% compared to the naive approach of adding zero skew constraints. Such amount of reduction may be unimportant for old technologies, but would yield precious power saving in today's power-hungry IC designs [14]. Therefore, an elaborated solution to the associative skew routing problem becomes increasingly necessary, especially for the aforementioned difficult instances. It is stated in [12]: "the key open issue is to find a heuristic that consistently outperforms greedy-DME for the domain with intermingled sink groups"

In this work, we will show that the associative skew routing problem can be solved simply by modifying existing bounded skew routing algorithm [4]. Experimental results on bench- mark circuits with the difficult instances show that our approach can reduce clock routing wirelength by 9%–15% compared to the extended greedy-DME method. In other words, we solved the key open issue raised in [12].

CHAPTER II

PROBLEM FORMULATION

In a clock tree, if clock signal delays to sink (flip-flop) a and b are t_a and t_b, respectively, the skew between them is defined as $t_a - t_b$. Given a set of clock sinks which are partitioned into k groups $G_1, G_2, ..., G_k$, each pair of sinks $s_{i,a}$ and $s_{i,b}$ in the same group G_i have a certain skew constraint but there is no skew constraint between any two sinks $s_{i,a}$ and $s_{j,b}$ which belong to two different groups G_i and G_j. For the simplicity of presentation, we let intra-group skew constraints be zero, i.e., clock signal delay to every sink in the same group has to be the same. Our method can be extended to non-zero prescribed skew or bounded skew constraint easily. The associative skew tree (**AST**) routing problem is to construct a clock tree such that the total wirelength is minimized while the intra-group skew constraints are satisfied.

Compared to conventional zero/prescribed skew routing, the formulation of **AST** implicitly leads to a by-product in addition to the clock tree itself. That is the skew among different sink groups which is not available in the input. The skew between group G_i and G_j is denoted as $S_{i,j}$. In [12], this inter-group skew is called offset. We need to specify the inter-group skew $S_{i,j}$ for all groups either implicitly or explicitly in solving the **AST** problem.

CHAPTER III

DELAY MODEL

In contrast to the previous work [12] which uses path length based metric, we employ the Elmore delay model as in many other clock routing works [1–6]. Although the Elmore delay is often inaccurate, it works very well for clock tree routing. One major reason of the inaccuracy is its neglection of the resistive shielding effect [15]. This is particularly significant when estimating the delay of a node near the source. However, the delay estimations in a clock tree are mainly for those far-end leaf nodes. Moreover, the error of *skew* estimation is usually very small despite large errors on *delay* estimation. In other words, the error in delay estimation is largely cancelled out in the subtraction operation when calculating skew. We have observed this phenomenon when we compare the Elmore based skew with SPICE simulation results.

CHAPTER IV

OBSERVATION

Why is the previous work [12] incapable of handling the instances where sink groups are intermingled? This is because that [12] constructs subtrees for each sink group separately and then stitches them together. Such approach may result in wire overlaps between subtrees of different groups despite the sophisticated stitching techniques [12]. In general, wire overlap implies inefficiency on wire usage. In Figure 2(a), two rectangle (circle) sinks from the same group are merged at node j (k). We use T_j and T_k to represent a subtree rooted at node j and k respectively. Then, subtree T_j and T_k are merged at the source node. The intermingling among different sink groups implies strong proximity connections among them. Therefore, constructing trees separately for each group would conflict with such strong connections. If we allow sinks from different groups to be merged as in Figure 2(b), the wirelength can be reduced up to 1/3 of its original wirelength.

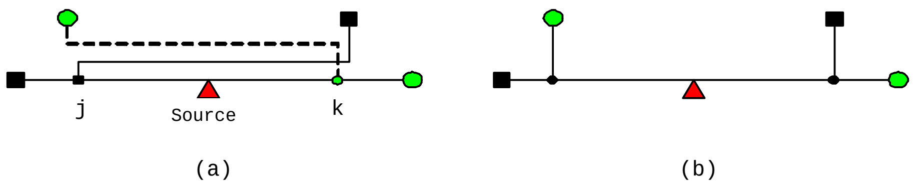

Fig. 2. Sink group is indicated by shape of sinks. Constructing trees for each group separately and then stitching them together as in (a) may result in wire overlap and large wirelength. Allowing merging between sinks from different groups as in (b) may reduce wirelength.

CHAPTER V

ALGORITHM

A. Overview and Merging Order

The observation of previous section tells that we should handle sinks from all groups simultaneously instead of separately in AST routing. This makes our AST construction more like traditional approaches [1–6] where the clock tree is constructed by successively merging subtrees in a bottom-up manner. Initially, each subtree is a single sink. Two subtrees can be merged to form a new subtree. The merging is repeated till only one subtree is left and this single subtree is connected to the clock source directly. Please note that this clock tree routing procedure is independent of the location of clock source.

The merging order in our approach is the minimum merging-cost merging scheme. Please note that we allow sinks from different groups to be merged simultaneously. This is the key difference from the work of [12].

B. Layout Embedding

When merging two subtrees, we need to determine the location of the root (merging node) of the new subtree formed from the merging. The location of the root needs to ensure that skews among sinks in the new subtree satisfy skew constraints. For the Elmore delay based clock routings, such location can be found by solving an equation that matches the delay of two subtrees [3]. The procedure of finding the location of merging nodes is called layout embedding.

Merging node can be a point, a segment or a polygon region according to skew constraints and routing algorithm. A famous layout embedding technique, **DME** (Deferred Merge Embedding) has a segment (along $\pm 45^o$ direction) instead of having a fixed single location without violating skew constrains. Such segment is called *merging segment* which is illustrated by the dashed lines in Figure 1(a). If the skew constraint is a bounded range instead of a single value, the location of a merging node can be anywhere in a specific polygon region to satisfy the skew bound. This region is called *merging region* in **BST** (Bounded Skew Tree) routing [4]. A merging region is constructed from two closest segments on the boundaries of its child merging regions. In Figure 1(b), merging regions are indicated by shaded area. In this paper, the merging segment and merging region corresponding node i are denoted as MS_i and MR_i, respectively.

After the bottom-up merging procedure, a merging segment tree (for **DME**) or a merging region tree (for **BST**) will be produced. Next, another top-down traversal of the tree is performed to decide the exact merging points on each merging segment or merging region such that the total wirelength is minimized. The top-down procedure in the proposed algorithm is the same as BST in [4]. However, bottom-up merging procedure is not the same as BST when we merge subtree nodes from different groups. In next sections, we will go over these different bottom-up procedures in detail.

C. Merging Subtrees from the Same Group

The scenario is almost the same as the classic DME or BST embedding according to skew constraints for the same group. If all sinks of a subtree T_a belong to a same group G_j, we say that this subtree is from group G_j. if the two subtree T_a and T_b to be merged are from the same group, we denote this fact as $T_a \bowtie T_b$. When skew constraint for the same group is zero, we can use the DME technique to find a single merging segment for them as shown in Figure 1(a). BST can be used to find the new merging region when skew constraint for the same group is a range rather than zero. Figure 1(b) shows how to construct the new merging region by BST.

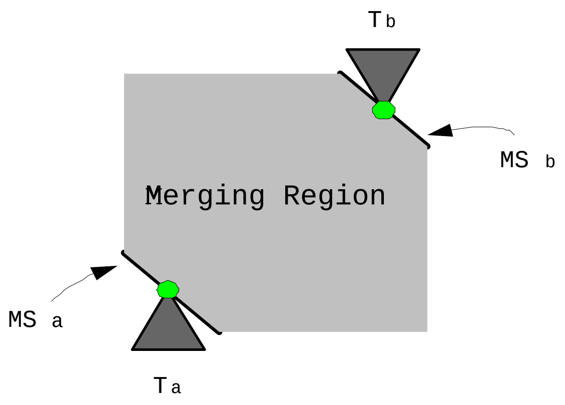

Fig. 3. If two subtrees, T_a and T_b belong to different groups, a merging region (shaded) is formed between MS_a and MS_b.

D. Merging Subtrees from Different Groups

This scenario is very similar to the BST routing [4]. Figure 3 shows simplest example of merging subtree from different groups. T_a and T_b are from group G_1 and G_2, respectively. Since there is no skew constraint between G_1 and G_2, there is no restriction for the location of their merging node. However, we prefer the wirelength from this merging to be minimal (which equals the Manhattan distance between MS_a and MS_b). Therefore, the shortest distance region (**SDR**) between MS_a and MS_b will be the merging region of T_a and T_b. Please note that this merging region implies a bounded range for skew $S_{2,1}$ between group G_1 and G_2. If two subtrees are from different groups and their root nodes are represented by merging regions, the merging of them is the same as that in BST.

E. Merging Subtrees from Partially Shared Groups

The scenario of merging subtrees from partially shared groups does not exist in either DME [1] or BST [4]. Besides, it is more complicated than the scenarios in previous two

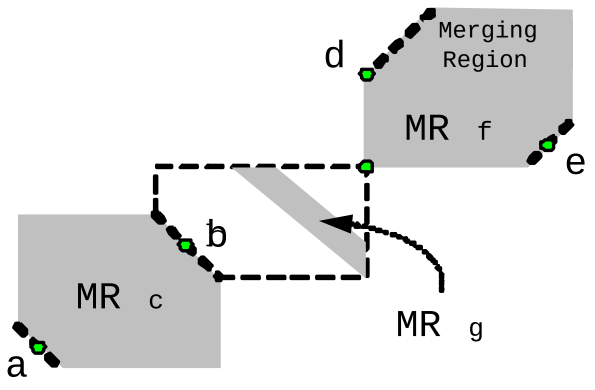

Fig. 4. If T_a and T_d are from the same group, a reduced merging region MR_g is formed when merging T_c and T_f.

subsections.

1. Instance 1

Consider the example in Figure 4 where subtree T_a and T_d are from group G_1, T_b is from group G_2 and T_e is from group G_3. Therefore, $T_a \bowtie T_d$. First, T_a and T_b are merged to form T_c and T_d and T_e are merged to form T_f. Now we are trying to merge T_c and T_f. Similar as BST [4], we choose the northeast boundary segment of merging region MR_c and the west boundary segment of merging region MR_f for this merging. The reason to choose them is that they are the closest boundaries between MR_c and MR_f. As both T_c and T_f contain sinks from group G_1, the merging between them has to satisfy the skew constraint within G_1. According to this skew constraint, we can find the merging region MR_g for this merging as shown in Figure 4. After this merging, G_1, G_2 and G_3 can be treated to form a new group $G_{1,2,3} = G_1 \cup G_2 \cup G_3$

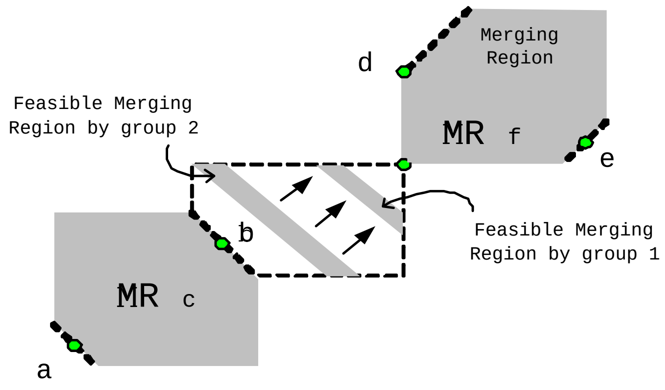
Merging Region
d
MR f
e
Feasible Merging Region by group 2
Feasible Merging Region by group 1
b
MR c
a

Fig. 5. If T_a & T_d and T_b & T_e are from the same group respectively, construct the feasible merging regions for each group and find an intersection between them. If there is no intersection, move one feasible merging region to another.

2. Instance 2

In Figure 5, both T_a and T_d are from group G_1, and T_b and T_e are from group G_2, i.e., $T_a \bowtie T_d$ and $T_b \bowtie T_e$. Similar as BST [4], we can construct feasible merging regions for G_1 and G_2 respectively. If the feasible merging region for G_1 and the feasible merging region for G_2 have an intersection between them, the intersected region will be the MR_g of T_c and T_f. However, if there is no intersected area between the feasible merging region of G_1 and G_2 like Figure 5, we need to move one area to another until there is an intersection. In this case, we move feasible merging region of G_2 to the feasible merging region of G_1. The movement of the feasible merging region of G_2 will result a wire sneaking [3]. Let us discuss the example shown in Figure 5 in detail. We are using elmore delay model to calculate delays. The root point separate the interconnection wire of the two subtrees into two halves (which may not equal). Each half wire segment is represented by π-model. Let α and β be the distance from the root of T_g to the root of T_c and T_f respectively. Let γ be the added wirelength to slide the feasible merging region of G_2. r and c are the unit resistance and capacitance repectively. C_i is the subtree capacitance of T_i. $l_{i,j}$ is the wirelength between the root of T_i and T_j. To ensure the delay from the root of T_g to the root of T_a and T_d being equal, it requires that

$$r\alpha(\frac{c\alpha}{2}+C_c)+rl_{ac}(\frac{cl_{ac}}{2}+C_a)=r\beta(\frac{c\beta}{2}+C_f)+rl_{df}(\frac{cl_{df}}{2}+C_d) \tag{5.1}$$

To ensure the delay from the root of T_g to the root of T_b and T_e being equal, it requires that

$$r\alpha(\frac{c\alpha}{2}+C_c)+rl_{bc}(\frac{cl_{bc}}{2}+C_b)=r\beta(\frac{c\beta}{2}+C_f)+r(\gamma+l_{ef})(\frac{c(\gamma+l_{ef})}{2}+C_e) \tag{5.2}$$

We need one more equation to solve for $\alpha, \beta, and \gamma$. It is

$$\alpha+\beta=l_{cf} \tag{5.3}$$

Hence, after solving Eq. (5.1), (5.2), (5.3), we find the location of the root of T_g and the wirelength that is needed for wire sneaking.

F. Enhancement on Merging Order

The basic AST-DME algorithm is outlined in Figure 6. In addition to the main ideas described here, this algorithm can be enhanced by two existing techniques.

1. Simultaneous multiple mergings for speed-up. It was pointed out in [2] that multiple subtree pairs can be merged simultaneously instead of mering only one pair each time. The multi-merging scheme can reduce the number of updatings on the nearest neighbor graph and therefore reduce the runtime.

2. Delay target based merging order for further wirelength reduction. It was observed in [6] that the wirelength may be affected by the relative delay targets of subtrees in addition to their proximity. A delay target of a node (or a subtree) is a desired delay from clock source to that node (or the root of the subtree). By merging subtrees with large delay targets first, the imbalance on delay targets of subtrees can be reduced.

```
Procedure: AST-DME
Input:  A set of sink groups $\mathcal{G} = \{G_1, G_2, ..., G_k\}$
        Skew constraints for sinks within each group
Output: A clock routing tree connecting all sinks and
        satisfying all intra-group skew constraints
1. Initialize a set $\mathcal{T}$ of subtrees with all sinks
2. While $|\mathcal{T}| > 1$
3.   Find a pair of subtrees $T_a \in \mathcal{T}$ and $T_b \in \mathcal{T}$ with min
     distance between their roots among all subtrees
4.   If both $T_a$ and $T_b$ are from $G_i$
       Merge $T_a$ and $T_b$ to $T_c$ at $MR_c$ satisfying skew
       constraints in $G_i$
5.   Else if $T_a$ and $T_b$ are from different groups
       Merge $T_a$ and $T_b$ to $T_c$ at $MR_c$ which is the
       SDR between $MR_a$ and $MR_b$
6.   Else if $T_a$ and $T_b$ share one group
       Merge based on nearest boundaries of $MR_a$
       and $MR_b$,
       Merge all sink groups involved with $T_a$ and $T_b$
7.   Else ($T_a$ and $T_b$ share multiple groups)
       Merge based on intersections of skew bounds
       induced by merging regions in $T_a$ and $T_b$,
       Merge all sink groups involved with $T_a$ and $T_b$
8.   $\mathcal{T} = \mathcal{T} - T_a - T_b + T_c$
```

Fig. 6. The proposed AST-DME algorithm.

Consequently, the chance of wire snaking is reduced.

Both of the above two techniques can be straightforwardly included in our AST-DME algorithm.

CHAPTER VI

EXPERIMENTAL RESULTS

Our algorithm is implemented in C/C++ and the experiments are performed on a Linux system with a Pentium-4 processor of 1.6GHz and 256MB RAM. The benchmark circuits are r1-r5 from [4].

In the first experiment, we divide each benchmark circuit space into rectangle boxes as many as the number of sink groups. If sinks are in the same rectangle space, they are in the same group. In other words, we created clusters of sink groups. We compare our AST-DME algorithm with EXT-BST algorithm [4]. For the case of associative skew routing, we simply set bounded skew range as 10ps and run the EXT-BST algorithm. The results are listed in Table I. As we expected, the wire reduction looks not so efficient because there is few chances to merge sinks in different groups. However, it results less wirelength than EXT-BST.

In the second experiment, we partition the sinks of each circuit into various number of groups which are intermingled with each other. We compare our AST-DME algorithm with EXT-BST algorithm. The results are listed in Table II. Both the EXT-BST and our AST-DME algorithms can satisfy the skew constraints. The results on wirelength show that our AST-DME consistently outperforms the EXT-BST algorithm. Usually, the improvement from AST-DME is more significant when the number of sink groups is increased and sink groups are intermingled.

The runtime of our algorithm is greater than that of EXT-BST as expected, but still at a reasonable order of magnitude.

Table I. Comparison between experimental results from an extended greedy-BST algorithm [4] and our algorithm of AST-DME with clusters of sink groups.

Circuit	#groups	Algorithm	Wirelen	Reduction	Maximum Skew(ps)	CPU(s)
r1 267 sinks	1	EXT-BST	1070421		10	25
	4	AST-DME	1048432	2.05%	49	25
	6	AST-DME	1041671	2.69%	53	25
	8	AST-DME	1040952	2.75%	57	26
	10	AST-DME	1039556	2.88%	60	26
r2 598 sinks	1	EXT-BST	2169791		10	74
	4	AST-DME	2112508	2.64%	39	75
	6	AST-DME	2112074	2.66%	46	75
	8	AST-DME	2093848	3.50%	56	75
	10	AST-DME	2091244	3.62%	62	76
r3 862 sinks	1	EXT-BST	2734959		10	94
	4	AST-DME	2664397	2.58%	45	96
	6	AST-DME	2647713	3.19%	63	98
	8	AST-DME	2644158	3.32%	67	98
	10	AST-DME	2646072	3.25%	66	98
r4 1903 sinks	1	EXT-BST	5442046		10	263
	4	AST-DME	5311981	2.39%	42	265
	6	AST-DME	5307627	2.47%	47	265
	8	AST-DME	5279328	2.99%	56	266
	10	AST-DME	5272254	3.12%	54	266
r5 3101 sinks	1	EXT-BST	8033650		10	407
	4	AST-DME	7836825	2.45%	49	409
	6	AST-DME	7799067	2.92%	53	409
	8	AST-DME	7771753	3.26%	55	409
	10	AST-DME	7754078	3.48%	61	410

Table II. Comparison between experimental results from an extended greedy-BST algorithm [4] and our algorithm of AST-DME with intermingled sink groups.

Circuit	#groups	Algorithm	Wirelen	Reduction	Maximum Skew(ps)	CPU(s)
r1 267 sinks	1	EXT-BST	1070421		10	25
	4	AST-DME	969872	9.39%	98	25
	6	AST-DME	945353	11.68%	107	25
	8	AST-DME	930384	13.08%	113	26
	10	AST-DME	926958	13.40%	121	26
r2 598 sinks	1	EXT-BST	2169791		10	74
	4	AST-DME	1940437	10.57%	78	77
	6	AST-DME	1938564	10.66%	93	77
	8	AST-DME	1865821	14.01%	117	79
	10	AST-DME	1855198	14.50%	119	79
r3 862 sinks	1	EXT-BST	2734959		10	94
	4	AST-DME	2452948	10.31%	89	97
	6	AST-DME	2371398	13.29%	132	98
	8	AST-DME	2386127	12.75%	128	101
	10	AST-DME	2379931	12.98%	137	101
r4 1903 sinks	1	EXT-BST	5442046		10	263
	4	AST-DME	4922763	9.54%	83	272
	6	AST-DME	4785931	12.06%	95	272
	8	AST-DME	4791754	11.95%	113	273
	10	AST-DME	4762357	12.49%	109	273
r5 3101 sinks	1	EXT-BST	8033650		10	407
	4	AST-DME	7247698	9.78%	98	411
	6	AST-DME	7094385	11.69%	107	412
	8	AST-DME	6984476	13.06%	111	412
	10	AST-DME	6915703	13.92%	122	413

CHAPTER VII

CONCLUSION

In this work, we attempt to solve the associative skew clock routing problem especially for the difficult instances where sink groups are intermingled. We find that this problem can be solved well by carefully assembling existing clock routing techniques. Experimental results show that our approach consistently outperforms an extension of a popular conventional method.

REFERENCES

[1] 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*, vol.39, no.11, pp.799–814, Nov. 1992.

[2] M. Edahiro, “A clustering-based optimization algorithm in zero-skew routings,” in *Proceedings of the ACM/IEEE Design Automation Conference*, Jul. 1993, pp.612–616.

[3] R.-S. Tsay, “Exact zero skew,” in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, Nov. 1991, pp.336–339.

[4] J. Cong, A. B. Kahng, C.-K. Koh, and C.-W. A. Tsao, “Bounded-skew clock and Steiner routing,” *ACM Transactions on Design Automation of Electronic Systems*, vol.3, no.3, pp.341–388, Jul. 1998.

[5] C.-W. A. Tsao and C.-K. Koh, “UST/DME: a clock tree router for general skew constraints,” in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, Nov. 2000, pp.400–405.

[6] R. Chaturvedi and J. Hu, “An efficient merging scheme for prescribed skew clock routing,” *IEEE Transactions on VLSI Systems*, vol.13, no.6, pp.750–754, Jun. 2005.

[7] J. G. Xi and W. W.-M. Dai, “Useful-skew clock routing with gate sizing for low power design,” *Journal of VLSI Signal Processing*, vol.16, no.2/3, pp.163–179, Jun./Jul. 1997.

[8] J. P. Fishburn, “Clock skew optimization,” *IEEE Transactions on Computers*, C-vol.39, pp.945–951, Jul. 1990.

[9] J. L. Neves and E. G. Friedman, "Design methodology for synthesizing clock distribution networks exploiting nonzero localized clock skew," *IEEE Transactions on VLSI Systems*, vol.4, no.2, pp.286–291, Jun. 1996.

[10] W.-C. D. Lam, C.-K. Koh, and C.-W. A. Tsao, "Power supply noise suppression via clock skew scheduling," in *Proceedings of the IEEE International Symposium on Quality Electronic Design*,March 2002, pp.355–360.

[11] I. S. Kourtev and E. G. Friedman, "Clock skew scheduling for improved reliability via quadratic programming," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, Nov. 1999, pp.239–243.

[12] Y. Chen, A. B. Kahng, G. Qu, and A. Zelikovsky, "On the associative-skew clock routing problem," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, Nov. 1999, pp.168–172.

[13] E. G. Friedman, "*Clock Distribution Networks in VLSI Circuits and Systems*," A Selected Reprint Volume, IEEE Press, Piscataway, NJ, 1995.

[14] T. Sakurai, "Perspectives on power-aware electronics," in *Proceedings of the IEEE International Solid-State Circuits Conference*, Feb. 2003, pp.26–29.

[15] J. Qian, S. Pullela, and L. T. Pillage, "Modeling the effective capacitance for the RC interconnect of CMOS gates," *IEEE Transactions on Computer-Aided Design*, vol.13, no.12, pp.1526–1535, Dec. 1994.

VITA

Min-seok Kim received his Bachelor of Science degree in electrical engineering from Texas A&M University at College Station in 2002. He entered the computer engineering program at Texas A&M University in January 2003. His research interest is the clock tree routing in VLSI design.

Mr Kim may be reached at Texas A&M University, Electrical Engineering Department, Mail Stop 3128, College Station, TX 77843-3128. His email address is aeromin@hotmail.com.