

# TSV-aware Topology Generation for 3D Clock Tree Synthesis

Wulong Liu<sup>1</sup>, Haixiao Du<sup>1</sup>, Yu Wang<sup>1</sup>, Yuchun Ma<sup>2</sup>, Yuan Xie<sup>3</sup>, Jinguo Quan<sup>4</sup>, Huazhong Yang<sup>1</sup>

<sup>1</sup>Dept. of E.E., TNLlist, Tsinghua Univ., Beijing, China <sup>2</sup>Dept. of C.S., TNLlist, Tsinghua Univ., Beijing, China

<sup>3</sup>Dept. of CSE, Pennsylvania State Univ., USA <sup>4</sup>Graduate School at Shenzhen, Tsinghua Univ., China

<sup>1</sup> Email: yu-wang@mail.tsinghua.edu.cn

**Abstract**—<sup>1</sup> Clock Tree Synthesis (CTS) mainly consists of two steps: 1) clock tree topology generation and 2) buffering and embedding. Due to the lack of the efficient model of TSVs, most previous CTS of 3D ICs ignore the effect of TSV planning in the first step. In this paper, we study the TSV-aware clock tree topology generation for 3D ICs by solving two major issues that the previous work has neglected: 1) the density distribution of allocated TSVs; 2) the parasitic and coupling effects induced by TSVs in constructing the topology of clock tree. The experimental results show that considering the impact of TSVs on 3D clock network in the topology generation step can meet the manufacture limitations and enable the designers to obtain the tradeoff among power consumption, the total wire length and the total number of TSVs. The experimental results show that TSVs number and power consumption can be reduced by up to 89.6% and 40.16% respectively with little variation of the total wirelength (the sum of total TSV equivalent wirelength and horizontal wire length) compared to the traditional NNG-based method. Besides, the mitigation of TSV-to-TSV coupling effect in 3D clock tree by implementing the proposed 3D CTS method is demonstrated in our experiment.

**Index Terms**—3D CTS, clock tree topology generation, TSV-to-TSV coupling.

## I. INTRODUCTION

TSV based 3D IC has attracted enough attention as a key technology to continue the scaling down trend predicted by the Moore's Law [1]. By 3D stacking, the average and maximum interconnect wire length can be significantly reduced, which further contributes to significant save on delay, power, area and cost [2]. The increasing interest of TSV based 3D IC design mainly relies on the slight modification of traditional EDA tools for 2D IC design. The lack of TSV-aware 3D IC physical design EDA tools, especially for a reliable 3D clock network synthesis tool, has hampered faster mainstream acceptance of 3D ICs [3]–[9].

For a 3D stacked IC, the clock tree distributes through the entire stacks and connects all the clock sinks in different layers by a single tree as shown in Figure 1. Different from 2D clock network, the clock signal distributes not only through X and Y direction, but also Z direction inter-connected by TSV. In the previous work of 3D Clock Tree Synthesis (CTS) [8]–[10], sinks in the entire stacks are just mapped onto one layer if satisfying the total TSV number constraint. Then the sink nodes with the smallest planar distance are paired with the highest priority. Meanwhile, one TSV is inserted between the paired sink nodes if they are located in different layers.

There are two major issues that hamper the actual implementation of previous 3D CTS work: one problem is the uneven distribution of TSVs in 3D clock network, which may not meet the manufacture limitations and leads to the reliability issue; the other is the neglect of the actual parasitic and coupling effects of TSVs in 3D clock topology generation, leading to more TSVs cost and power consumption.

<sup>1</sup>This work was supported by IBM, National Science and Technology Major Project (2010ZX01030-001-001-04, 2011ZX01035-001-001-002) and National Natural Science Foundation of China (No. 61028006, 61261160501), and Tsinghua University Initiative Scientific Research Program.

For the first problem, the previous 3D CTS methods have large possibility to generate 3D clock network with uneven distribution of TSVs, especially for the case shown in Figure 2. From CPU benchmarks in ISPD contest 2010 [11], we find that clock sinks are unevenly distributed on the chip with high density areas (for logic) and low density areas (for memory or IP core). When these benchmarks are folded into 3D stacks, two high density areas may be stacked together. By implementing previous 3D CTS methods, TSVs will be concentrated in these high density areas, as Figure 3 (a) shows. **This uneven distribution of TSVs brings many problems for 3D ICs design, such as low yield, high power, high cost and the violation of manufacture limitations as shown in Figure 1.** It tends to be ideal if the number of TSVs is reduced between two logic areas located in different layers. From this point of view as Figure 3 (b) shows, we propose a sorting algorithm to cluster sink nodes with high density to decrease the impact on 3D chip caused by TSV insertion.

The other major problem of the previous work is the neglect of the actual parasitic and coupling effects of TSVs during the clock topology generation phase, which increases the actual latency and power. Based on the 45nm Predictive Technology Model [12], the latency of one TSV equals to that of a 8.6 $\mu$ m interconnect wire in Elmore model with the same driving load (*per unit resistance and capacitance are 0.1 $\Omega$ / $\mu$ m and 0.2fF/ $\mu$ m, while the TSV capacitance and resistance are 28ff and 0.053 $\Omega$  respectively* [13]). **The parasitic effect of TSVs will take a significant part of latency between two neighbor nodes with the scaling down trend.** If the latency of TSV is neglected in 3D clock topology generation phase, the final 3D clock network could not be optimized with much more overhead of TSVs number, power and delay. This paper presents an efficient way to judge the detailed insertion of one TSV by finding the nearest paired sink nodes in actual. The parasitic effect induced by TSVs is considered in constructing the topology of 3D clock tree. **In addition, the coupling effect between two adjacent TSVs in 3D ICs could be very problematic because of the big size of TSVs, which could lead to the increase of path delay and power consumption, and**

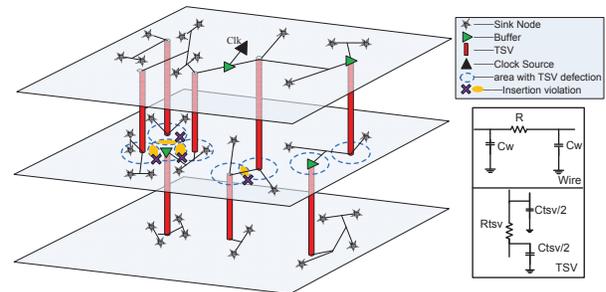


Fig. 1. Three-die stacked 3D clock tree and TSVs inserted violations marked with yellow circles

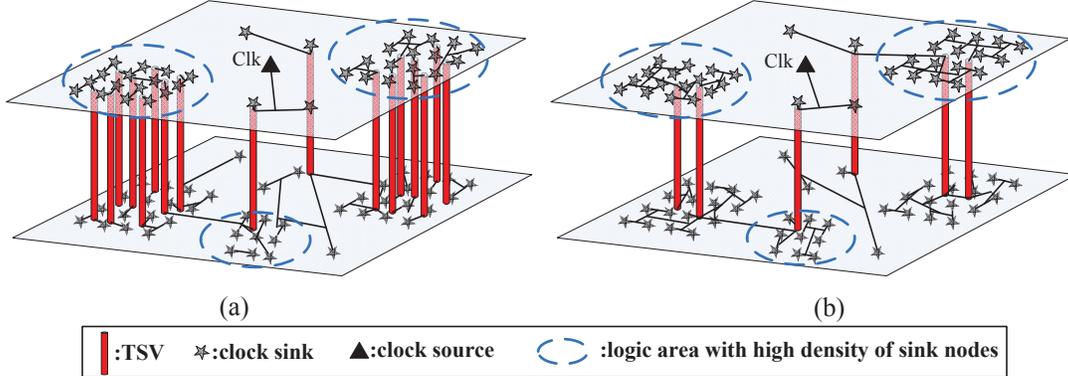


Fig. 3. (a) The concentration of TSVs caused by stacking two logic areas between two dies; (b) The ideal distribution of TSVs between two dies in clock network.

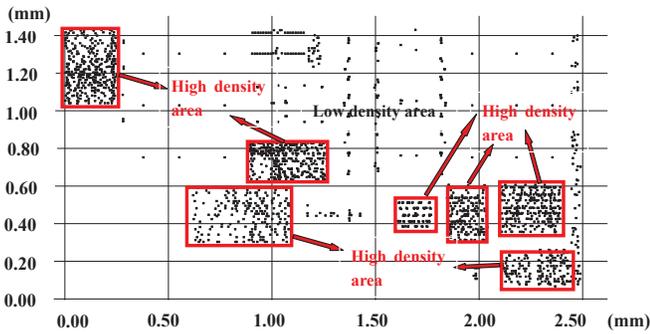


Fig. 2. Clock sinks distribution of CPU benchmark in ISPD contest 2010 (07)

**also the logic and timing violation.** The detailed coupling model between two adjacent TSVs will be presented in Section III. In this work, our proposed density-aware sorting algorithm can mitigate the coupling effect on 3D clock network by extending the distance between adjacent TSVs.

The contributions of this paper are as follows.

- We propose a density-aware sorting algorithm to cluster sink nodes with high density. This approach can achieve a balanced distribution of TSVs, guarantee the manufacture limitations, and enhance the reliability especially for low power 3D clock network design.
- We present a concept named TSV equivalent wire length (TEWL) to evaluate the overhead of TSVs in clock tree topology generation step. Besides, the TEWL together with the traditional horizontal wire length are utilized to evaluate the overall overhead of 3D clock network. By considering the TEWL during the 3D clock topology generation phase, we can achieve up to 8.8% and 4.2% reduction of power consumption and total TSVs number respectively but with little variation of the total wirelength (the sum of total TEWL and horizontal wire length) compared with the traditional NNG-based method.
- Combining our proposed clustering algorithm with TEWL considered in clock topology generation phase, our 3D CTS method can enable the designers to obtain the tradeoff among the power consumption, the total wire length and the number of TSVs. The simulation results show that the number of TSV and power consumption can be reduced by up to 89.6% and 40.16%

respectively with little variation of the total wirelength (the sum of total TEWL and horizontal wire length) compared with the traditional NNG-based method.

- According to our knowledge, we are the first to analyze TSV-to-TSV coupling effect in 3D clock network. The experimental results demonstrate that our 3D CTS method can mitigate this effect in 3D clock network.

This paper is organized as follows. Section II reviews the related work. Section III gives the detailed TSV-to-TSV coupling model considered in this work. Section IV shows the problem formulation. Section V presents our proposed sorting algorithm. Section VI explains the details of our 3D CTS method. Experimental results are shown and discussed in Section VII. And we conclude the paper and present our future work in Section VIII.

## II. RELATED WORK

In the last few years, several researchers have conducted fruitful researches on 3D clock design, which mainly focus on zero (bounded) skew aware 3D clock network design [5], slew aware 3D clock network design [8], low power 3D clock network design [9], robust aware 3D clock network design [6], [7] and testable aware 3D clock network design [4], [14]. Zhao et al. [14] and Kim et al. [4] focused on clock tree synthesis with pre-bond testability for 3D stacked IC design.

For 3D abstract tree topology generation, there are three existing algorithms 3D-MMM [6], [8], [9], MMM-3D [3] and NNG-based algorithm (Nearest Neighbor Graph) [10]. Both of 3D-MMM and MMM-3D are based on 2D MMM (method of means and medians) algorithm proposed in [15]. They partition a subset of sinks vertically and horizontally according to the given TSV number threshold but not the TSVs density. Besides, NNG-based 3D clock topology generation method is based on the slightly modification of the traditional NNG method [16].

For low power aware 3D CTS design, [8], [9] propose power and slew-aware clock network design for TSV based 3D ICs, which mainly consists of two major steps: 1) 3D abstract tree generation and 2) slew-aware buffering and embedding. Kim et al. [3] presents a 3D embedding method to reduce the total wire length without detailed experimental results about the skew or slew. The low power target is achieved by the reduction of wirelength with much more TSVs overhead. Power consumption of TSVs is not optimized in the previous work.

However, all the above work do not consider the density distribution of TSVs. In addition, TSV's parasitic and coupling effects are neglected when generating the clock tree topology. In [10], the authors made an effort to take the latency and even distribution of TSVs into account. However, the latency of TSV is just simply included in the merging cost, and no experimental result demonstrated this approach can satisfy the manufacture limitations. Therefore, this paper mainly proposes a novel TSV-aware 3D CTS method taking the actual effect induced by TSVs insertion into account.

### III. TSV-TO-TSV COUPLING MODEL

In 3D ICs, the coupling effect between two adjacent TSVs could be very problematic because of the big size of TSVs. This TSV-to-TSV coupling could lead to the increase of path delay, power, and the logic and timing violation. In this work, we simplify the equivalent lumped circuit model of two coupled TSVs proposed in [17] as shown in Figure 4. We use the following equations presented in [17], [18] to compute the capacitances and the resistances:

$$C_{TSV} = \frac{1}{4} \frac{2\pi\epsilon_0\epsilon_r}{\ln\left(\frac{r_{TSV} + t_{OX}}{r_{TSV}}\right)} \times l_{TSV} \quad (1)$$

$$C_{si} = \epsilon_0\epsilon_{si} \frac{2(r_{TSV} + t_{OX}) + \alpha}{d} \times l_{TSV} \quad (2)$$

$$C_{Bump} = \frac{\epsilon_0\epsilon_r}{d - 2r_{Bump}} \times \pi \times r_{Bump} \times l_{Bump} \quad (3)$$

where  $\epsilon_0$  and  $\epsilon_{si}$  are the dielectric constant in vacuum and silicon,  $\alpha$  is the scaling factor,  $r_{TSV}$  and  $l_{TSV}$  are the TSV radius and height respectively,  $r_{Bump}$  and  $l_{Bump}$  are the radius and the height of a bump,  $t_{OX}$  is the thickness of the insulator and  $d$  is the distance between two TSVs. In order to explore the latency induced by one TSV considering the coupling effect, we apply one pulse signal to the TSV and simulate the equivalent lumped circuit model using SPICE by using the parameters defined in [18]. The simulation result approves that the latency during stimulus signal through one TSV can be reduced by 65% if the distance between two adjacent TSVs extending from 11um to 100um as shown in Figure 5, because of the mitigation of TSV-to-TSV coupling. This kind of TSV-to-TSV coupling induced latency changes the equivalent wire length of TSVs, and will be bad in the 3D clock network design which has rigid timing requirements. In this work, we will utilize a sorting algorithm to mitigate TSV-to-TSV coupling effect and improve the nonuniform distribution of TSVs.

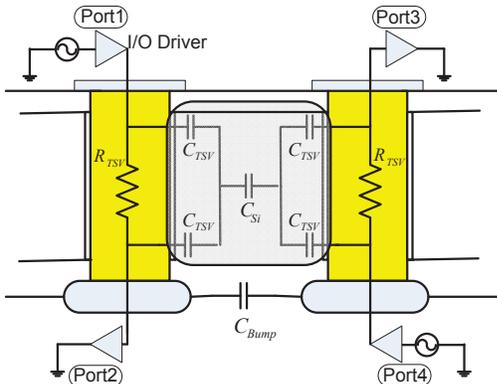


Fig. 4. TSV-to-TSV coupling model

### IV. PROBLEM FORMULATION AND OUR SOLUTION FLOW

The TSV-aware clock tree topology generation problem for 3D IC can be formulated as following: Given a set of sinks distributed in all dies and the parasitic of interconnect wires, buffers and TSVs, the goal of our 3D CTS method is to generate a 3D clock tree topology that connects all the clock sinks in different layers by a single tree as shown in Figure 1 and achieves the following objective:

$$\text{Minimize : } f(r) = \alpha f_{wire\_length}(r) + \beta f_{TSV\_num}(r) + \gamma f_{power}(r) \quad (4)$$

in which

$$f_{wire\_length}(r) = \sum_{i \in \{0,1,\dots,N\}} (l_i) + \sum_{i \in \{0,1,\dots,M\}} (TEWL_i) \quad (5)$$

$$f_{power}(r) = f_{TSV\_power}(r) + f_{buffer\_power}(r) + f_{wire\_power}(r) \quad (6)$$

and subject to

$$\alpha + \beta + \gamma = 1, (\alpha \geq 0, \beta \geq 0, \gamma \geq 0) \quad (7)$$

$$r_{TSV} \leq r \leq r_{bench} \quad (8)$$

where  $f_{wire\_length}(r)$ ,  $f_{TSV\_num}(r)$ ,  $f_{power}(r)$  are the total wire length including the TEWL, the total number of TSVs, and the total power consumption respectively,  $\sum_{i \in \{0,1,\dots,N\}} (l_i)$  is the total wire length of the 3D clock tree,  $\sum_{i \in \{0,1,\dots,M\}} (TEWL_i)$  is the total equivalent wire length of TSVs,  $f_{TSV\_power}(r)$ ,  $f_{buffer\_power}(r)$ ,  $f_{wire\_power}(r)$  are the power consumption of TSVs parasitic and coupling, buffer and wire, and  $\alpha$ ,  $\beta$ ,  $\gamma$  are the weights to make a tradeoff among the total number of TSVs, the total wire length and the total power consumption.  $\alpha$ ,  $\beta$ ,  $\gamma$  are decided by designers requirement. Besides, Formula 8 constrains the region of sorting radius  $r$ , which can be adjustable according to the actual manufacture and design limitations.

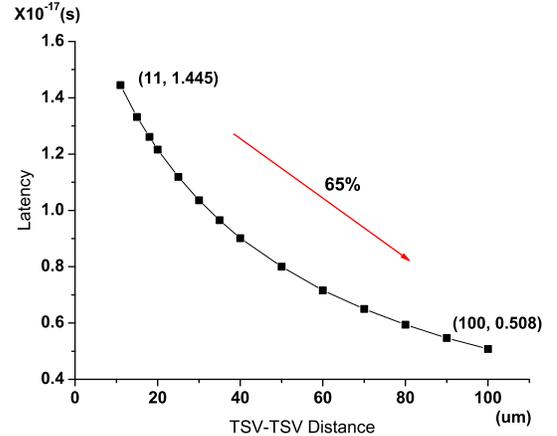


Fig. 5. The latency varies with different distance between two TSVs

This work mainly focuses on TSV-aware clock tree topology generation for 3D ICs, which is implemented by three major steps as Table II shows. At the first step, density-aware sorting algorithm is implemented to cluster sink nodes with high density. Then, we modify the traditional defer merging embedding (DME) [19] and the nearest neighbor graph(NNG) [10] methods to generate the clock

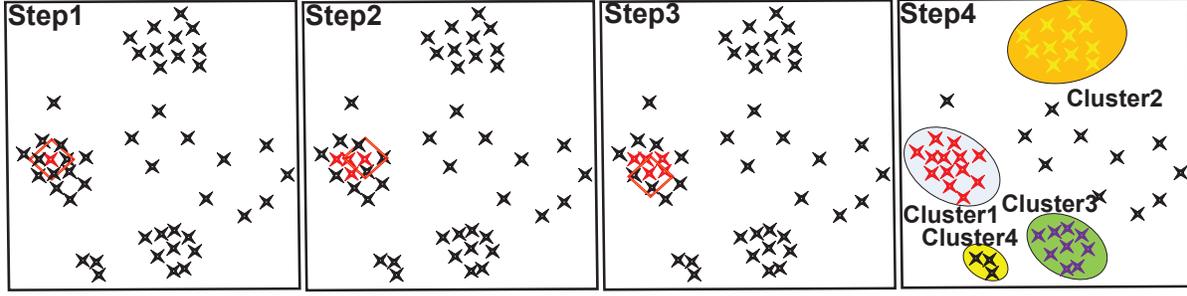


Fig. 6. The detailed steps of our proposed sorting algorithm to cluster each high density area in a growing method

TABLE I  
PARAMETERS DEFINITION

	Parameters definition
$r$	Defined as the minimum permitted Manhattan distance between two TSVs, and also utilized as the sorting radius
$TEWL$	Defined as the equivalent wire length for TSV
$pair(i, j)$	The paired nodes with the minimum Manhattan distance
$Incr(i, j)$	The increased wire length and TEWL after merging the $pair(i, j)$
$sink(i).layer$	The layer number of $sink(i)$
$Sink_N(k)$	Sink node in N-th layer
$Dist(i, j)$	The Manhattan distance between $Sink(i)$ and $sink(j)$
$Dens(v)$	The nearest neighbors of $sink(v)$
$class(i)$	Equals to a sorted cluster
$sort$	The total number of sorted clusters
$Neighbor(i)$	The neighbor number of $sink(i)$
$sort\_constraint$	The boundary of sink number to identify dense or sparse
$r_{TSV}$	The diameter of one TSV
$r_{bench}$	The radius of one benchmark

tree topology for each sorted cluster. At the second step, together with the unsorted sink nodes at the first step, the roots of sub-trees constructed in the first step are constructed into the whole topology of 3D clock tree. At the last step, in order to explore the contribution of our 3D clock tree topology method to the final 3D clock network, we also implement the traditional buffering and embedding method [6] to determine exact geometric locations for all nodes, TSVs, and buffers in a top-down fashion.

TABLE II  
OUR DETAILED SOLUTION FLOW

Detailed Solution Flow
<b>Step1:</b> Sorting and construct the clock topology for each sorted cluster
<b>Substep1:</b> Density aware sorting
<b>Substep2:</b> Construct the clock topology for each sorted cluster utilizing modified DME
<b>Step2:</b> Construct the whole 3D clock topology
<b>Substep1:</b> Mapping unsorted sink nodes and sorted root nodes onto one layer
<b>Substep2:</b> Paring the nearest neighbor nodes by considering TEWL
<b>Step3:</b> Top-down buffering and embedding

## V. SORTING AND CONSTRUCTION OF THE CLOCK TOPOLOGY FOR EACH SORTED CLUSTER

### A. Density-aware sorting algorithm

Density based sorting algorithms are mainly utilized in Data Mining and Image Processing [20]. As a representation of all kinds of sorting algorithms, DBSCAN [20] can discover any form of clusters with noise disturbance. In this work, based on the concept of DBSCAN, we propose a simplified sorting algorithm to cluster the areas with high density of sink nodes, avoiding the inserted violations of adjacent TSVs as Figure 7 shows. The pseudo-code is shown in Figure 7.

Figure 6 shows an example to illustrate the details of our sorting algorithm about how to cluster each high density area with growing

method. In step1, the number of each sink node's neighbors within a certain Manhattan radius  $r$  is calculated. Then, one of sink nodes is selected as the center. If it has two or more neighbors within a certain Manhattan radius, meaning that the TSV could not insert among this node and its neighbors, all the neighbor nodes are marked and put into a cluster. In step2, other node selected from the marked cluster is set as the center instead to judge whether it has closer neighbors. Its neighbors are also marked and put into this cluster as step2. After  $n$  times iterations ( $n$  is the total number of sink nodes), all the sinks that close to each other can be sorted into one cluster (step3). If no sink node can be found to add into this cluster, it will break this iteration and return out all the sink nodes within this cluster. Then, after the next  $n$  times iterations, another area that have high density of sink nodes can be sorted into their corresponding clusters. All areas with high density of sink nodes can be clustered as shown in Figure 6(step4).

Once the minimum distance that can avoid the impact on the CMOS logic caused by TSVs insertion is given, our sorting algorithm can cluster areas with high density of sink nodes in each layer to avoid inserted violations of adjacent TSVs. The minimum distance can be obtained from experiments like in [21]. In this work, in order to explore the distributed density of clock sinks for different benchmarks, we set a series of different values for the minimum Manhattan distance  $r$  for each benchmark, and the sorted results are shown in Table III.

Simplified density based sorting algorithm	
Input:	the coordinates $C(x, y, z)$ for each node in $G$ , a certain Manhattan radius $r$ , and the boundary of total number to identify dense or sparse
Output:	some sorts with their owned sinks
1	<b>for</b> each sink node $v$ in $G$
2	{ given a certain Manhattan radius $r$ ;
3	$N \leftarrow Sink(v).layer$ ;
4	$Sink_N(k) \in \{Sink_N(1), Sink_N(2) \dots Sink_N(N_d)\}$ ;
5	<b>if</b> ( $Dist(v, k) < r$ )
6	{
7	Put $Sink_N(k)$ into $Dens(v)$ ;
8	}
9	}
10	<b>for</b> each sink node $v$ in $G$
11	{ $class(v) \leftarrow Dens(v)$ ;
12	$Sink(i) \in class(v)$ ;
13	<b>do</b>
14	{ <b>if</b> ( $Neighbor(i) > sort\_constraint$ )
15	{
16	Mark all the sinks within $Den(i)$ ;
17	}
18	} <b>while</b> (there are unmarked nodes )
19	Put all the marked nodes into a sort( $Num$ );
20	$Num++$ ;
21	}

Fig. 7. The pseudo-code of our proposed simplified density based sorting algorithm

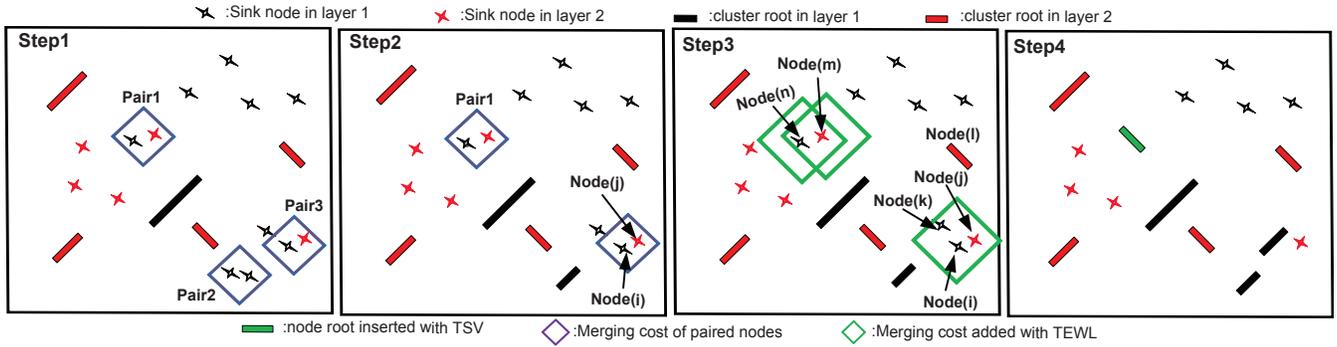


Fig. 8. An example to illustrate the procedures of constructing the whole 3D clock topology

TABLE III  
SINK NODES SORTED RESULTS FOR DIFFERENT BENCHMARKS

Benchmark	sinks	$R(50\mu m)$	$R(50\mu m)$	$R(10\mu m)$	$R(10\mu m)$
		Num_sort	Sorted sinks	Num_sort	Sorted sinks
ISPD01	1107	8	118	0	0
ISPD02	2249	41	296	22	72
ISPD03	1200	62	1200	176	881
ISPD04	1845	123	1682	68	271
ISPD05	1016	101	966	91	370
ISPD06	981	35	952	47	198
ISPD07	1915	46	1854	127	476
ISPD08	1134	33	1055	92	429

### B. Construct the clock topology for each sorted cluster

It is to note that all sink nodes marked with the same cluster are located in the same layer. Then, the modified DME method is utilized to generate a 2D clock sub-tree topology for each sorted cluster, and the root node of the sub-tree in each cluster is also generated. The root node together with other internal nodes in each cluster are marked with the same layer label. Meanwhile, the original sink nodes in each cluster are marked with having been merged. Figure 9 presents an example of constructing abstract tree for each sorted cluster.

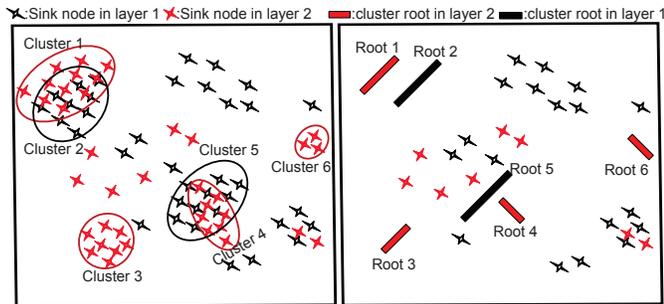


Fig. 9. Construct the clock topology for each sorted cluster, node in different layers marked with different colours

## VI. CONSTRUCT THE WHOLE 3D CLOCK NETWORK

### A. Construct the whole 3D clock topology

Figure 8 shows an example with detailed procedures to construct the whole 3D clock topology. Once the topology of the subtree in each cluster has been constructed, we can utilize our modified 3D DME method to construct the whole topology of 3D clock tree. In this work we propose an efficient way to judge whether one TSV is needed between two nodes by taking into account the latency induced

by TSV insertion. In order to make a uniform criterion with the traditional total wire length, we propose a TSV equivalent wire length concept (TEWL). TEWL equals to the length of a wire with the same latency calculated according Formula 9 with assumption that the TSV and the wire have the same driving load.

$$RC_L + \frac{RC}{2} = \frac{rcTEWL^2}{2} + rTEWLC_L \quad (9)$$

where  $R$  and  $C$  are the resistance and capacitance of TSV.  $r$  and  $c$  are the per unit resistance and capacitance of wire. When the load capacitance  $C_L$  is smaller than that of TSV, the latency of one TSV equals to a wire with the length TEWL, and TEWL can be easily calculated according to the above formula. Based on the 45nm Predictive Technology Model [12] (per unit resistance and capacitance are  $0.1\Omega/\mu m$  and  $0.2fF/\mu m$ ), it is frequently assumed that TSV diameter is  $4\mu m$ , TSV capacitance is  $28ff$ , and resistance is  $0.053\Omega$  [13], which leads to the latency that equals to a interconnect wire with a length about  $8.6\mu m$ .

Firstly, all the unsorted sink nodes and the root node of each sorted cluster are mapped onto one layer, which also means that their  $Z$  coordinates are neglected.

Next, we implement the custom bucket partitioning [19] approach to find the nearest neighbor for each node by constructing the nearest neighbor graph. Those nodes with minimum Manhattan distance are paired temporarily, such as pair1, pair2 and pair3 in Figure 8. There will be two cases for these paired sink nodes as the following: 1) the paired sink nodes locate in the same layer like pair2; 2) the paired sink nodes locate in the different layers like pair1 and pair3. For the first case, it will be no problem to merge these two nodes in pair2 by implementing the DME method, and the traditional merging cost which defines the increased wire length by merging these two paired nodes can be calculated. For the second case, the previous work [7]–[9] only marks that one TSV is needed to interconnect these two paired nodes located in the different layers. However, because of the latency induced by TSV insertion, the paired nodes may not be with the real minimum merging cost. In this work, in order to consider the overhead induced by TSV insertion, the sum of TEWL and horizontal wire length is set as the criterion to judge whether the paired nodes are with the minimum merging cost.

For the second case discussed above, if one of the paired nodes having marked with a TSV interconnected, the Manhattan distance between the paired nodes is compared with the defined minimum sorting radius  $r$ . When it is smaller than  $r$ , the original paired nodes are abandoned to avoid the violation of two adjacent TSVs. When it is larger than  $r$ , the increased wire length  $Incr(i, j)$  is

calculated by merging the paired nodes  $pair(i, j)$  located in different layers at first by considering  $TEWL$ . Then, one step of bucket partitioning is implemented again just for each node in the pair and those nodes located in the same layer with  $Node(i)$ . Supposed that the new nearest neighbor of  $Node(i)$  is  $Node(k)$ , the increased wire length  $Incr(i, k)$  induced by merging the paired nodes  $pair(i, k)$  is calculated. The same procedure will be implemented for  $Node(j)$ , and its nearest neighbor node  $Node(l)$  located in the same layer is found. The increased latency overhead  $Incr(j, l)$  is also compared with  $Incr(i, j)$ . If  $Incr(i, j)$  for the previous paired sink nodes located in different layers is smaller than both of  $Incr(i, k)$  and  $Incr(j, l)$ . It means that the original paired nodes in  $pair(i, j)$  are the real nearest sink nodes and can be merged, or this pair of nodes will not be merged leading to no TSV insertion.

In this work, the total downstream wirelength (the sum of horizontal wire length and TEWL) for each child node is compared and the TSV is inserted in the edge between the merged parent node and that child node with less total wire length plus TEWL. Besides, for simplification, we suppose that the TSV is just inserted below the merged parent node.

It is to note that our proposed TSV based 3D CTS method may generate an imbalanced tree structure because of the cluster based 2D subtree generation. However, our approach can guarantee that the downstream capacitance and latency are balanced when finding the paired nodes. Thus, the skew can be well improved.

### B. Buffering and embedding

In order to explore the contribution of our 3D clock tree topology method to the final 3D clock network, we also modified the traditional slew-aware buffering and embedding method [22] to determine exact geometric locations for all nodes, TSVs, and buffers in a top-down fashion. This is a highly efficient dynamic programming based algorithm to optimally solve slew-aware buffering with discrete buffering locations. Because of space limitation, we will not describe the details of buffering and embedding.

## VII. SIMULATION AND DISCUSSION

### A. Simulation setup

In our simulation, we first construct a 3D clock tree topology by our proposed TSV-aware approach shown in Section V under the density constraint and TSV's parasitic. This part of work is implemented using C++ and achieves the results about horizontal wirelength, the number of TSVs, TEWL and buffer number. Then the exact netlist of the entire 3D clock network is extracted for simulation in Spice, and outputs the detailed results about power, delay and skew. The technical parameters utilized in our work are based on 45nm Predictive Technology Model (PTM) [12]: the unit-length wire resistance is  $0.1\Omega/\mu\text{m}$ , and the unit-length wire capacitance is  $0.2\text{fF}/\mu\text{m}$ . The parasitic resistance and capacitance of a TSV are  $0.035\Omega$  and  $28\text{fF}$ . The ISPD clock network contest benchmarks [11] are utilized for our experiments under Elmore model. Since these benchmarks are originally designed for 2D ICs, we divide these benchmarks into four layers and we shrunk each layer so that the total area of four layers for 3D IC equals to that of the original one layer for 2D IC. In addition, we set the maximum slew value is 200ps when implementing slew-aware buffering.

### B. Analysis of our sorting algorithm

1) *Impact on the density distribution of TSVs*: In order to explore the impact of our sorting algorithm on the density distribution of TSVs, we simply divide a benchmark into some small square blocks

and the number of TSVs in each block is counted. Taking the benchmark named  $ISPD10\_07$  and  $ISPD10\_08$  for example, per square length is set as  $45\mu\text{m} \times 45\mu\text{m}$ . The total number of TSVs is 598 and 65 with unsorted and sorted ( $r=50\mu\text{m}$ ) situations for Benchmark  $ISPD10\_07$ , and the maximum TSV density in per square is 16 and 11 respectively when unsorted, while reducing to 3 after implementing our sorting algorithm as shown in Figure 10. Therefore, these results demonstrate that our approach can achieve a balanced distribution of TSVs, and satisfy the manufacture limitations.

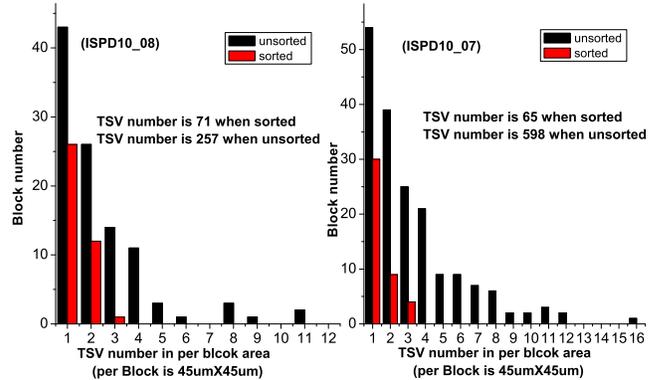


Fig. 10. The TSV density distribution when sorted and unsorted

TABLE IV  
IMPACT BY THE VARIATION OF MINIMUM MANHATTAN RADIUS  $r$  ( $ISPD10\_01$ ),  $N\_TSV$  is the total number of TSVs;  $N\_Bufs$  is the total number of buffers;  $WL$  is the total horizontal wirelength

$r(\mu\text{m})$	$N\_TSV$	$WL(\mu\text{m})$	$N\_Bufs$	Skew(ns)	Power(mW)
0(NNG)	659	169901	566	0.521	77.89
50	615	175286	556	0.573	77.20
100	575	179463	559	0.561	76.57
150	476	191920	520	0.431	72.89
200	287	225659	505	0.472	69.50
250	136	286390	555	0.639	72.67
300	48	333904	602	0.586	76.73

2) *The sorting results with different constraint of minimum Manhattan radius*: One contribution of our work is to utilize a density-aware sorting algorithm to circle area with high density of sink nodes when generating the topology of 3D clock tree. In order to explore the tradeoff among the total number of TSVs, wirelength and power by implementing our sorting algorithm, we vary the minimum Manhattan radius  $r$  with different values when implementing our proposed sorting algorithm at the step of constructing the topology of 3D clock tree. By taking experiments for different benchmarks, the results show that most cases have the similar characteristics as shown in Table IV. Note that it will be the traditional NNG-based methods if the minimum Manhattan radius  $r$  is set as 0um. The experimental results demonstrate that our proposed clustering algorithm contributes to obtain the tradeoff among power, the number of TSVs and horizontal wirelength. For example, when  $r$  ranges from 0um to 200um, the power consumption can be reduced by up to 10.8% and the TSV numbers can be reduced by up to 56.5% but with about 32.81% horizontal wirelength overhead. When  $r$  is larger than 200um, the number of TSVs will also be decreased, but power consumption and wirelength will increase rapidly. The reason is that the number of TSVs and buffers decrease with the increase of sorting

TABLE V

SIMULATION RESULTS WHEN CONSIDERING TSV PARASITIC AND COUPLING EFFECTS COMPARED WITH NNG-BASED 3D CTS METHODS, ( $N_{TSV}$  is the total number of TSVs;  $N_{Bufs}$  is the total number of buffers;  $TEWL$  is the TSV equivalent wire length;  $Tot\_WL$  is the sum of  $TEWL$  and  $WL$ )

Benchmark	Traditional NNG-based 3D CTS					Our 3D CTS method only considering TSV effect					Reduction Ratio		
	$N_{TSV}$	( $TEWL+WL$ ) Tot_WL(um)	$N_{Bufs}$	Power (mW)	Skew (ns)	$N_{TSV}$	( $TEWL+WL$ ) Tot_WL(um)	$N_{Bufs}$	Power (mW)	Skew (ns)	$N_{TSV}$ (%)	Power (%)	Tot_WL (%)
ISPD10_01	659	20175+169901	566	77.89	0.573	631	19440+170322	551	76.58	0.5329	4.2	1.7	5.4
ISPD10_02	1482	36990+288503	1029	142	0.543	1475	36540+286474	1015	140.81	0.6443	0.5	0.8	0.7
ISPD10_03	374	8010+18113	221	31.49	0.595	341	7170+18329	210	30.18	0.5983	8.8	4.2	2.4
ISPD10_04	822	24600+59251	418	60.09	0.747	785	23550+58649	391	57.98	0.6994	4.5	3.5	2.0
ISPD10_05	152	3795+32727	152	16.20	0.449	155	3750+32584	115	16.11	0.4791	2.0	0.5	9.7
ISPD10_06	412	11730+22758	218	31.95	0.710	416	11340+22200	218	31.34	0.547	1.0	1.9	2.7
ISPD10_07	598	10860+44625	282	40.5	0.556	568	10350+44551	278	39.71	0.637	5.0	2.0	1.0
ISPD10_08	257	5640+31043	183	25.61	0.546	254	5505+30417	178	25.23	0.674	1.2	1.5	2.0

radius  $r$ , while the wirelength will increase. Therefore, there exists the tradeoff among power, horizontal wirelength and the number of TSVs.

3) *Impact of considering TSVs parasitic effect in clock tree topology generation:* In order to explore the impact of TSVs parasitic on 3D clock network, we present the comparison of the results when only TSVs parasitic effects are considered in topology generation phase with the traditional NNG-based 3D clock topology generation method for different benchmarks. The results are shown in Table V. The simulation results show that we can achieve up to 8.8% reduction of TSVs and about 4.2% reduction of power, and the sum of total  $TEWL$  and wirelength ( $Tot\_WL$ ) can be reduced by 0.7% to 9.7% compared with the traditional NNG-based method. These results also demonstrate that the neglect of TSVs parasitic effect is not reasonable.

4) *Analysis of our proposed 3D CTS method without considering TSV-to-TSV coupling effect:* In order to explore the impact of our proposed 3D CTS method implementing sorting and considering TSVs parasitic effect, we present the comparison of the number of TSVs, the number of buffers, wirelength, power, skew with the traditional NNG-based 3D clock topology generation method for different benchmarks. The experimental results are shown in Table VI. For different benchmarks, the optimal sorting radius  $r$  may be different because of the variation of their originally sink nodes distribution. In this work, we set the sorting radius as 50um for the same criterion that could be appropriate for most of the selected benchmarks. The experimental results show that our 3D CTS method can achieve up to 89.6% reduction of TSVs and 40.16% reduction of power but with little variation of the total wirelength compared with the traditional NNG-based method. At the last of this section, we will discuss the reason that we can achieve the significant power reduction but with little variation of the total wirelength.

Note that for ISPD2010\_01 and ISPD2010\_02, our method is not that effective, because these two benchmarks have relatively more evenly distributed sink nodes comparing with other benchmarks. All above discussed results are obtained without considering TSV-to-TSV coupling effect. It is apparent that our 3D CTS method can achieve better performance when adding TSV-to-TSV coupling model in our simulation platform, which will be analyzed in the next subsection. In addition, all the above skew values appear to be a little bit large because of the slew-aware buffering, and we do not implement any skew-aware optimization method for simplification. Our future work will focus on the skew-aware buffering and optimization.

5) *Analysis of our proposed 3D CTS method considering TSV-to-TSV coupling effect:* In order to analyze the TSV-to-TSV coupling effect on 3D clock tree, TSV-to-TSV coupling model presented in Section III is implemented in 3D clock tree. According to our

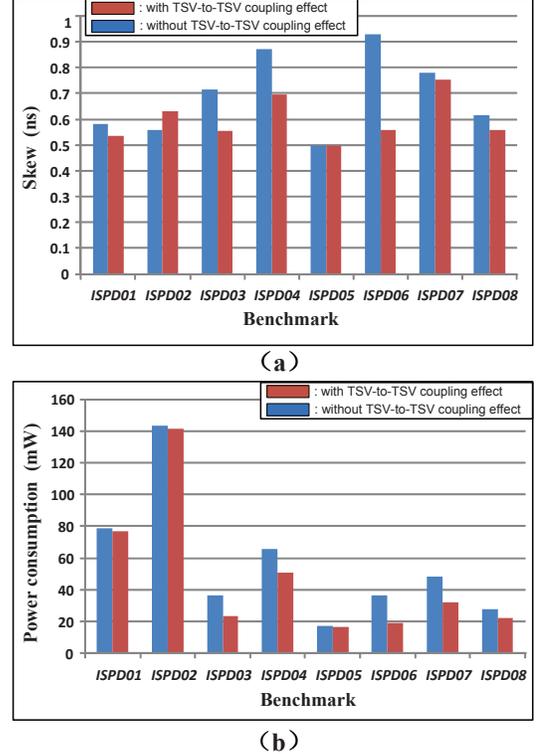


Fig. 11. Our skew and power consumption optimization results with and without TSV-to-TSV coupling effect in 3D clock network

simulation result in Figure 5, TSV-to-TSV coupling effect can be neglected when the distance between two adjacent TSVs is larger than 100um. Therefore, in order to decrease the simulation time, we just add the TSV-to-TSV coupling spice model in the 3D clock tree spice network when the distance between two adjacent TSVs is smaller than 100um. The simulation results show that we can achieve much more skew and power consumption optimization after considering TSV-to-TSV coupling effect for most cases as shown in Figure 11 (b). One interesting thing is that the skew may not be decreased after adding TSV-to-TSV coupling effect for the benchmark ISPD02 as shown in Figure 11 (a). It is mainly the reason that the TSV-to-TSV coupling effect may increase much more delay of the critical path compared with the shortest path, which increases the clock skew.

6) *Analysis of our power reduction:* The reason that we can achieve the significant power reduction but with little variation of the total wirelength can be explained by a general example as shown in Figure 12. We assume three cases with similar total wirelength (the sum of wire length and  $TEWL$ ): (a) a wire without TSV; (b)

TABLE VI

SIMULATION RESULTS OF OUR PROPOSED 3D CTS COMPARED WITH NNG BASED 3D CTS METHODS, ( $N_{TSV}$  is total number of TSVs;  $N_{Bufs}$  is the total number of buffers;  $TEWL$  is the TSV equivalent wire length;  $Tot\_WL$  is the sum of  $TEWL$  and  $WL$ )

Benchmark	Traditional NNG-based 3D CTS					Our proposed 3D CTS method					Reduction Ratio		
	$N_{TSV}$	(TEWL+WL) Tot_WL(um)	$N_{Bufs}$	Power (mW)	Skew (ns)	$N_{TSV}$	(TEWL+WL) Tot_WL(um)	$N_{Bufs}$	Power (mW)	Skew (ns)	$N_{TSV}$ (%)	Power (%)	Tot_WL (%)
ISPD10_01	659	20175+169901	566	77.89	0.573	606	18990+175633	554	76.58	0.5988	8.0	1.68	-2.4
ISPD10_02	1482	36990+288503	1029	142	0.543	1379	34380+291094	1003	140.28	0.75	7.0	1.21	0.005
ISPD10_03	374	8010+18113	221	31.49	0.595	50	1230+24438	154	23.37	0.5136	86.5	25.78	1.7
ISPD10_04	822	24600+59251	418	60.09	0.747	163	4395+94483	239	48.71	0.6594	80.1	18.94	-17
ISPD10_05	152	3795+32727	152	16.20	0.449	70	1815+36004	96	15.90	0.4791	53.9	1.85	-3.5
ISPD10_06	412	11730+22758	218	31.95	0.710	43	1080+32134	122	19.12	0.5804	89.6	40.16	3.7
ISPD10_07	598	10860+44625	282	40.5	0.556	61	1290+66079	216	31.79	0.6249	89.6	21.50	-21
ISPD10_08	257	5640+31043	183	25.61	0.546	51	1080+43697	147	22.10	0.5418	80.2	13.71	-22

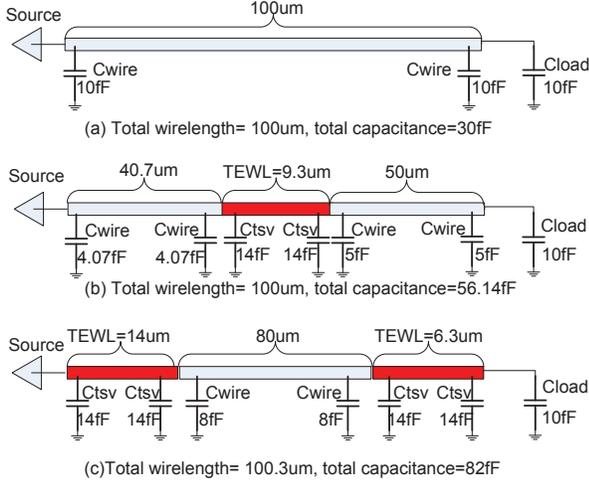


Fig. 12. One example to explain our power reduction with three different cases

a wire inserted with one TSV; (c) a wire inserted with two TSVs. Each TEWL is calculated according to our TEWL model in Formula 9. The results show that case (c) has the maximum total capacitance ( $case(a) < case(b) < case(c)$ ), meaning that case (a) will have the minimum power consumption. Our 3D CTS method will result in a majority of cases like case (a) or case (b) due to the reduction of the number of TSVs. This also demonstrates our 3D CTS method can achieve significant power reduction by reducing the number of TSVs, but with little variation of the total wirelength.

## VIII. CONCLUSION

In this paper, we propose a novel TSV-aware 3D clock tree topology generation method while taking the actual density constraint of TSVs and their parasitic and coupling effects into account. The experimental results demonstrate that our density based sorting approach can guarantee the manufacture limitations of TSVs and enable the designers to obtain the tradeoff among power consumption, the total wire length and the total number of TSVs. Our simulation show that the number of TSVs and power consumption can be reduced by up to 89.6% and 40.16% respectively with little variation of the total wirelength compared with the traditional NNG-based method. Besides, we can achieve much more skew and power consumption optimization after including TSV-to-TSV coupling effect.

## REFERENCES

[1] Y.-F. Tsar, Y. Xie, N. Vijaykrishnan, and M. J. Irwin, "Three-dimensional cache design exploration using 3dcacti," in *Proceedings of the IEEE International Conference on Computer Design*, 2005, pp. pp. 519–524.

[2] X. Dong and Y. Xie, "System-level cost analysis and design exploration for 3d ics," in *ASP-DAC*, Jan. 2009.

[3] T.-Y. Kim and T. Kim, "Clock tree embedding for 3d ics," in *ASP-DAC*, 2010, pp. 486–491.

[4] —, "Clock tree synthesis with pre-bond testability for 3d stacked ic designs," in *DAC*, 2010, pp. 723–728.

[5] —, "Bounded skew clock routing for 3d stacked ic designs: Enabling trade-offs between power and clock skew," in *Green Computing Conference 2010*, pp. 525–532.

[6] J. Minz, X. Zhao, and S. K. Lim, "Buffered clock tree synthesis for 3d ics under thermal variations," in *ASPDAC 2008*, pp. 504–509.

[7] M. Mondal, A. Ricketts, S. Kirolos, T. Ragheb, G. Link, N. Vijaykrishnan, and Y. Massoud, "Thermally robust clocking schemes for 3d integrated circuits," in *DATE 2007*, pp. 1–6.

[8] X. Zhao and S. K. Lim, "Power and slew-aware clock network design for through-silicon-via (tsv) based 3d ics," in *ASP-DAC*, 2010, pp. 175–180.

[9] X. Zhao, J. Minz, and S. K. Lim, "Low-power and reliable clock network design for through-silicon via (tsv) based 3d ics," *Components, Packaging and Manufacturing Technology, IEEE Transactions on*, vol. 1, no. 2, pp. 247–259, 2011.

[10] T.-Y. Kim and T. Kim, "Clock tree synthesis for tsv-based 3d ic designs," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 16, pp. 48:1–48:21, October 2011.

[11] "Isprd contest 2010 [online]. available: <http://archive.sigda.org/isprd/contests/10/isprd10cns.html>," 2010.

[12] "Predictive technology model [online]. available: <http://ptm.asu.edu/>."

[13] J.-S. Yang, J. Pak, X. Zhao, S. K. Lim, and D. Pan, "Robust clock tree synthesis with timing yield optimization for 3d-ics," in *ASP-DAC*, 2011, pp. 621–626.

[14] X. Zhao, D. Lewis, H.-H. Lee, and S. K. Lim, "Pre-bond testable low-power clock tree design for 3d stacked ics," in *ICCAD 2009*, 2009, pp. 184–190.

[15] S. A. Jacson, M.A.B. and E. S. Kuh, "Clock routing for high-performance ics," in *DAC 1990*, 1990, pp. pp. 573–579.

[16] M. Eda, "A clustering-based optimization algorithm in zero-skew routing," in *DAC 1993*, 1993, pp. pp. 612–616.

[17] C. Liu, T. Song, J. Cho, J. Kim, J. Kim, and S. K. Lim, "Full-chip tsv-to-tsv coupling analysis and optimization in 3d ic," in *DAC 2011*, June 2011, pp. 783–788.

[18] T. Song, C. Liu, D. H. Kim, S. K. Lim, J. Cho, J. Kim, J. S. Pak, S. Ahn, J. Kim, and K. Yoon, "Analysis of tsv-to-tsv coupling with high-impedance termination in 3d ics," in *ISQED 2011*, March 2011, pp. 1–7.

[19] D. J. H. Huang, A. B. Kahng, and C.-W. A. Tsao, "On the bounded-skew clock and steiner routing problems," in *DAC*, 1995, pp. 508–513.

[20] Ester, Matin, and K. H. P. and et al, "A density based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the 2nd International conference on Knowledge Discovery and Data Mining*, 2006, Portland, Oregon.

[21] H. Chaabouni and M. e. a. Rousseau, "Investigation on tsv impact on 65nm cmos devices and circuits," in *IEDM*, 2010, pp. 35.1.1–35.1.4.

[22] S. Hu, C. Alpert, and e. a. Hu, J., "Fast algorithms for slew constrained minimum cost buffering," in *DAC 2006*, 2006, pp. pp. 308–313.