Through-Silicon-Via-Induced Obstacle-Aware Clock Tree Synthesis for 3D ICs

Xin Zhao and Sung Kyu Lim
School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, GA 30332, U.S.A.
{xinzhao, limsk}@ece.gatech.edu

***Abstract*— In this paper, we present an obstacle-aware clock tree synthesis method for through-silicon-via (TSV)-based 3D ICs. A unique aspect of this problem lies in the fact that various types of TSVs become obstacles during 3D clock routing including signal, power/ground, and clock TSVs. Some of these TSVs become placement obstacles, *i.e.*, they interfere with clock buffers and clock TSVs; while other TSVs become routing obstacles, *i.e.*, clock wires cannot route through them. Thus, the key is to perform TSV-induced obstacle-aware 3D clock routing under the following goals: (1) clock TSVs and clock buffers are located while avoiding overlap with placement obstacles; (2) clock wires are routed while avoiding routing obstacles; and (3) clock skew and slew constraints are satisfied. Related experiments show that our TSV-obstacle-aware clock tree does not sacrifice wirelength or clock power too much while avoiding various TSV-induced obstacles.**

I. Introduction

Three dimensional ICs (3D ICs) have prominent properties to achieve higher integration and further miniaturization. Designers are able to achieve many benefits by using TSVs, such as shorter wirelength, less wire delay, lower power, and smaller chip area. Recent 3D research has focused on improving performance, lowering power consumption, increasing reliability and manufacturability, and designing testing schemes.

Despite the gains of TSV-based 3D ICs, TSVs create serious blockages for 3D clock routing. TSVs are vertical vias through the silicon die, and provide die-to-die communication for multiple functional nets, such as power and ground networks, clock networks, and signal nets. As shown in Figure 1.a), three kinds of TSVs co-exist in 3D designs. Power and ground TSVs (=P/G TSVs) usually have large diameter, and utilize many local vias to provide the vertical connection in between; signal TSVs and clock TSVs occupy silicon area and have relatively smaller diameter compared with P/G TSVs. Before clock tree synthesis, P/G TSVs and signal TSVs are inserted and occupy both silicon and metal space. Figure 1.b) shows the TSV diameters in terms of the standard cell row height in $45nm$ technology. One can easily see that TSVs are significant layout obstacles due to their large size compared with logic gates and local wires. The TSV-to-gate size ratio is predicted to increase in ITRS 2009, especially when the keep-out-zone around TSVs is taken into account. Clock routing in 3D IC becomes challenging because these various types of TSVs all become obstacles.

Existing work on 3D clock tree synthesis focuses on thermal-aware clock skew minimization [1], wirelength and power minimization [2], [3], pre-bond testability [4], [5]. But, none of these works take into account TSV-induced obstacles.

In existing works on obstacle-aware clock routing, Kahng and Tsao [6] proposed deferred merging and embedding (DME)-based obstacle expansion rules to determine feasible embedding locations for the internal nodes. In [7], Kim and Zhou presented a planar obstacle-aware routing scheme to clean up overlaps between clock nets and obstacles. Huang *et al.* [8] proposed another DME-based clock routing method to avoid obstacles with the help of a track graph. These works mainly focus on routing-obstacle avoidance, *i.e,* to prevent clock nets from crossing over the given obstacles. In addition, there are several works on avoiding insertion of clock buffers on the given blockages based on either maze routing [9] [10], or breadth-first-search [11]. However, none of these work can directly solve the TSV-obstacles in 3D clock tree construction problem.

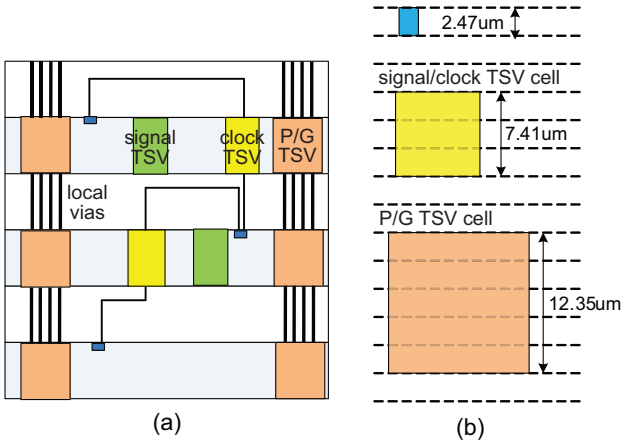

Fig. 1. Side and top-down view of via-first power/ground (P/G) TSVs, clock TSVs and signal TSVs. (a) P/G TSVs use many local vias in between vertically, (b) size of the TSV cells (= TSV + keep-out-zone) in terms of the standard cell row height (45nm technology).

In this paper, we focus on solving a practical 3D clock routing problem that stems from TSV-induced obstacles. Our contributions are summarized as follows. First, we analyze that the P/G TSVs and signal TSVs perform as two different types of obstacles in 3D clock routing. Different from traditional obstacle-aware clock routing, the special properties of TSV-induced obstacles are: 1) Placement obstacles and routing obstacles co-exist during 3D clock routing; 2) Clock buffers and clock TSVs occupy significant area and should be considered during clock routing to avoid overlap with other TSV obstacles; 3) TSV obstacles are distributed over the entire 3D layout area. Second, we develop a TSV-induced obstacle-aware DME policy to construct a TSV-overlap-free buffered clock tree. We extend the traditional concept of merging segment to represent clock TSV and clock buffer insertion; we develop *Expanded-Obstacle Cutting* and *Nine-Region-Based Cutting* techniques to determine overlap-free merging segments; we present two detour policies to handle clock routing in heavily crowded regions. We apply our algorithm on several real benchmarks and show the efficiency of our algorithm. Related experiments show that our TSV-obstacle-aware clock tree does not sacrifice wirelength or clock power too much while avoiding

This material is based upon the work supported by the National Science Foundation under Grant No. CCF-0917000, Semiconductor Research Corporation (SRC ICSS), and the Interconnect Focus Center (IFC).

978-1-4673-0772-7/12/$31.00 ©2012 IEEE

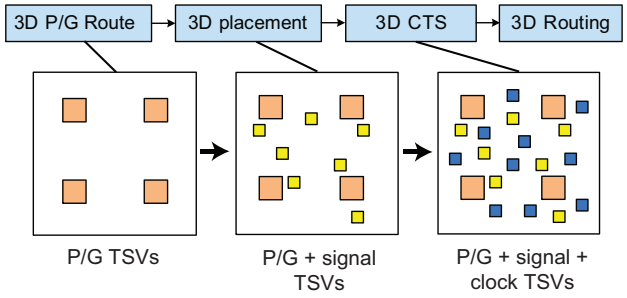

Fig. 2. Addition of TSVs during 3D IC physical design. Note that P/G and signal TSVs are added before clock routing.

various TSV-induced obstacles.

II. TSV Obstacle Analysis

The 3D IC physical design flow consists of several steps. In each design stage, different types of TSVs are added. We use a TSV map to illustrate the size and location of TSVs. Figure 2 shows how the TSV map evolves during each 3D design stage. During 3D power planning, we construct the 3D power/ground network. Power and ground TSVs (= P/G TSVs) are inserted at regular locations. To obtain small resistance, P/G TSVs may have a larger size than other TSVs, occupy several standard cell rows, and utilize many local vias to provide the vertical connection in between. During 3D placement, we determine the location of gates as well as signal TSVs. The major reason is that we are able to assign enough space for the signal TSVs, which are several times larger than gates; otherwise, inserting signal TSVs during routing would create many problems. These P/G and signal TSVs then become obstacles during clock routing[1]. In addition, clock TSVs and buffers are added during clock routing, where clock TSVs themselves become another source of TSV-induced obstacles[2]. These TSV obstacles behave in the following ways:

- Signal TSVs: they occupy silicon area only, and work as placement obstacles for clock buffers and clock TSVs. This means that, 1) clock TSVs and clock buffers are not allowed to overlap with existing signal TSVs; 2) clock nets are *allowed* to routed over the signal TSVs because their landing pads are in M1 and free up the metal spaces above. Figure 3.a) shows an illustration.
- P/G TSVs: they occupy both silicon area and metal layers, and thus function as both placement and routing obstacles. This means that, 1) clock TSVs and clock buffers should avoid overlap with existing P/G TSVs; 2) the clock net is not allowed to route over the P/G TSV. Figure 3.b) shows an illustration.
- Clock TSVs/buffers: besides P/G TSVs and signal TSVs, 3D clock tree synthesis itself also inserts clock buffers and clock TSVs. They become the same kind of clock routing obstacles as signal TSVs if added in an iterative fashion such as DME-based clock tree embedding.

[1]In large 3D IC design, 3D global clock synthesis may be performed after floorplanning, where signal TSVs have not been inserted yet. As a result, the TSV-obstacles for 3D clock synthesis include P/G TSVs (acting as both placement and routing obstacles), clock TSVs, and clock buffers. To show the efficiency of our algorithm, this paper focuses on the design flow where 3D clock routing performs after placement, where all types of TSV-induced obstacles exist.

[2]We focus on inserting clock TSVs during clock routing to gain shorter wirelength and lower power. Preserving TSVs before routing is an alternative solution, but out of the scope of this work.

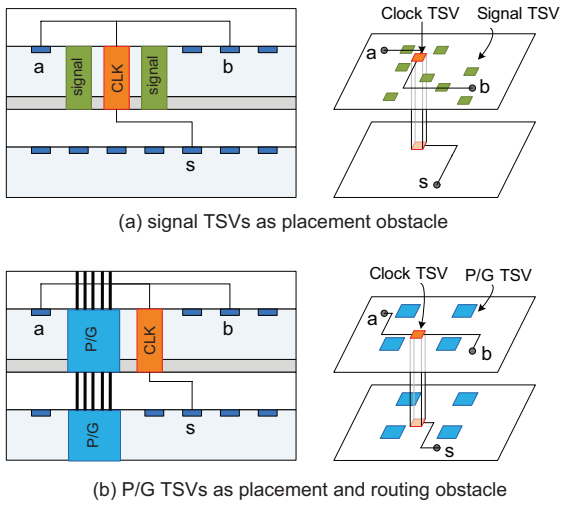

Fig. 3. TSV-induced obstacles in 3D clock routing for clock sinks a, b and s. (a) signal TSVs as placement obstacles, where the clock net is allowed to route over the signal TSVs, (b) P/G TSVs as placement and routing obstacles, where clock net is not allowed to route over the P/G TSVs.

Due to the sheer size of TSVs, detour policies are required to handle the cases when TSV obstacles significantly block the routing and placement area.

III. Preliminaries

A. Problem Formulation

The formal definition of *TSV-induced obstacle-aware 3D clock routing problem* is as follows: Given a 3D TSV obstacle map consisting of signal TSVs and P/G TSVs on each die, a set of clock sinks on each die, dimensions of clock buffers and clock TSVs, an upper bound on TSV count, the objective is to construct an overlap-free buffered 3D clock tree such that 1) clock skew is zero; 2) clock wirelength and power are minimized; 3) clock slew is bound under the given constraint. *The overlap-free constraint* requires that 1) clock buffers and clock TSVs do not overlap with the signal TSVs and P/G TSVs; 2) clock nets are not routed over the P/G TSVs.

B. Extension of Merging Segment Concept

In our 3D TSV-obstacle-aware clock routing, we extend the concept of merging segment (ms) that is primarily used for clock internal nodes only to denote the candidate locations of non-zero-sized clock buffers and clock TSVs under minimum skew and wirelength objectives. Specifically, $msp(p)$ and $msc(t)$ denote the ms of an internal clock node p and the center of non-zero-sized clock TSV or buffer, respectively. Figure 4 shows an illustration of the extended merging segment concept.

We focus on via-first TSV-induced obstacles in 3D clock tree synthesis[3]. These obstacles can be classified into two types: *placement obstacle* that blocks the silicon area and affects clock buffer/clock TSV insertion. This comes from P/G TSVs, signal TSVs, and clock TSVs; *Routing obstacle* that blocks the routing area and affects the clock routing topology. This comes from P/G TSVs. We use the following kinds of merging segments in this work:

- *Placement-overlap-free merging segments*: collection of the merging points that the corresponding non-zero-sized clock components, *i.e.*, clock TSVs and clock buffers, have no overlap

[3]We apply our TSV-obstacle-aware clock routing on via-first TSV 3D application. It can be easily extend to via-middle or via-last TSVs.

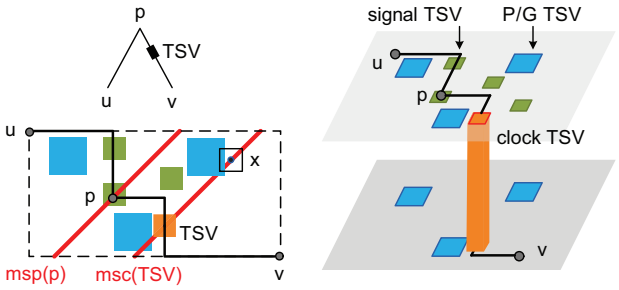

Fig. 4. Illustration of the extended merging segment concept. When merging nodes u and v in different dies, $msp(p)$ denotes the merging segment of node p; $msc(TSV)$ denotes the center-point locations of the clock TSV. Signal TSVs allow node p and the clock net to route over it. However, clock TSV x cannot overlap with a P/G TSV.

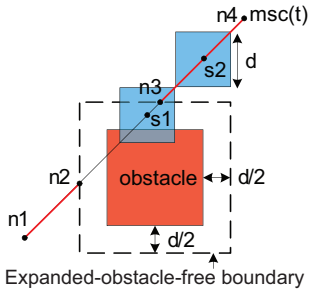

Fig. 5. Expanded-Obstacle Cutting on a merging segment $msc(t)$. The expanded-overlap-free boundary determines that segment $n1$-$n2$ and $n3$-$n4$ are the feasible merging segments. A clock TSV with $s1$ as the center will cause an overlap with the obstacle, whereas inserting the TSV with its center on $s2$ is safe.

with the placement obstacles and have its center point located along the msc.

- *Routing-overlap-free merging segments*: potential location of msc and msp, which are able to reach the children merging segments with the minimum distance while avoiding routing obstacles.
- *Feasible merging segments*: For msp, the feasible merging segment becomes the routing-overlap-free merging segments; for msc, the feasible merging segment satisfies both placement-overlap-free and routing-overlap-free requirements.

IV. Overview of the Algorithm

Our TSV-obstacle-aware clock routing algorithm consists of the following two steps:

Bottom-up feasible merging segment construction: Our goal is to determine the feasible merging segments (= FMS) for internal nodes, clock buffers, and clock TSVs. Depending on the merging types (*e.g.,* wire merging, TSV merging), the flow is different. When merging $ms(u)$ and $ms(v)$ to $msp(p)$, we first generate the $msp(p)$ under a zero-skew constraint and determine its FMS using the *Nine-Region-Based Cutting* method explained in Section V-B. Note that clock TSVs and clock buffers may be inserted along the edges (u, p) or (v, p) together. If a clock TSV is required on (v, p) because u and v are in different dies, we aim to find the FMS for p and the TSV. When a clock buffer is required to be inserted along (u, p) or (v, p), our goal is to determine the FMS for p using the *Nine-Region-Based Cutting* method and for the buffer and TSV using both the *Nine-Region-Based Cutting* and *Expanded-Obstacle Cutting* method explained in Section V-B and Section V-A. If no FMS can be found in the merging area with the shortest distance, we utilize two detour policies for both placement obstacles and routing obstacles explained in Section VI.

Top-down obstacle-aware embedding: Our goal is to decide the exact embedding point along the FMS, and to determine the clock routing topology. The embedding points for the clock buffers and clock TSVs should avoid overlap between other clock TSVs, P/G TSVs, signal TSVs, and clock buffers. In addition, we use the *Nine-Region-Based* method to determine the final routing-overlap-free topology.

V. Feasible Merging Segments

We present two techniques to obtain overlap-free merging segments: *Expanded-Obstacle Cutting* to obtain a placement-overlap-free merging segment, and *Nine-Region-Based Cutting* to determine a routing-overlap-free merging segment. Based on whether a given TSV is a placement or routing obstacle, we apply different cutting policies. For P/G TSVs, both the *Expanded-Obstacle Cutting* and *Nine-Region-Based Cutting* methods are used; for signal TSVs, only the *Expanded-Obstacle Cutting* method is used.

A. Expanded-Obstacle Cutting

The goal of *Expanded-Obstacle Cutting* method is to determine the placement-overlap-free merging segment of clock TSVs and buffers. The outcome is an insertion of a clock TSV or a clock buffer with the center point located along the merging segment. In this case, this clock TSV or the clock buffer should have no overlap with other placement obstacles such as P/G TSVs, signal TSVs, other clock TSVs, and clock buffers that are already existing in the layout.

Given an initial merging segment $msc(t)$, the dimension of the clock component, d, to be added (= either clock TSV or clock buffer), and a placement obstacle $obst$, the basic procedure of *Expanded-Obstacle Cutting* is as follows: We first construct an expanded-overlap-free boundary (EOFB) by expanding the $obst$ by the distance of $d/2$ in all four directions. We then utilize EOFB to determine a feasible merging segment: any merging point along the $msc(t)$ outside the EOFB is a placement-overlap-free point; in other words, any point along $msc(t)$ inside the EOFB will have overlap with the placement obstacle. Figure 5 shows an illustration.

B. Nine-Region-Based Cutting

Given a merging segment of a child u, our goal in *Nine-Region-Based Cutting* is to find the routing-overlap-free feasible merging segments of its parent p (= either an internal clock tree node, clock TSV, or clock buffer), so that the merging segment of p provides a feasible routing topology to its child u with the shortest distance.

A routing obstacle (in red) partitions the routing area into nine regions with its four extended boundary lines as shown in Figure 6.a). These nine regions are used to determine the connectivity between a pair of nodes. For instance, node p can connect to u in both HV (horizontal first, then vertical) and VH (vertical first, then horizaontal) topology, whereas VH routing type from nodes p' to u' is blocked by the obstacle. Note that these nine regions are symmetric; they can be classified into three groups: Group A: Regions 1, 3, 7, and 9; Group B: Regions 2, 4, 6, and 8; and Group C: Region 5. The connectivity of merging segments can be easily determined by referring to the region groups. (1) When p is located in Region 2, it is two-way (both HV and VH) connectable to Regions 1, 2 and 3; HV connectable to Regions 4, 6, 7 and 9; and is not connectable to Regions 5 and 8. The same discussion applies to the case when p is located in Regions 4, 6, or 8. (2) When p is located in Region 1, it is two-way (both HV and VH) connectable to Regions 1, 2, 3, 4, 7 and 9; HV connectable to Region 6; VH connectable to Region 8; and is not connectable to

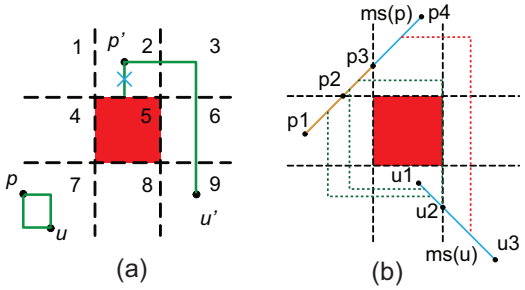

Fig. 6. Nine-Region-Based Cutting method. (a) Nine regions partitioned by a routing obstacle in red. p to u is HV and VH connectable, and p' to u' is HV only. (b) $(p1, p2)$ and $(p2, p3)$ are the routing-overlap-free merging segment of $ms(p)$ to its child $ms(u)$, $(p3, p4)$ is not due to the shortest distance constraint.

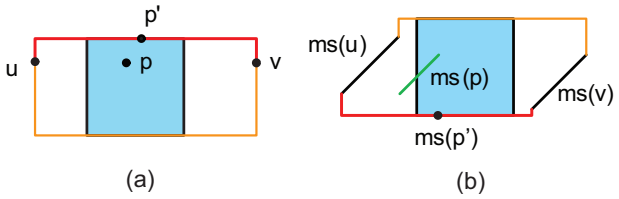

Fig. 7. Detour policy when a routing-obstacle blocks the routing region. (a) merging segment for u and v are points, where the top (= red) detour is chosen over the bottom (= orange), (b) merging segments for u and v are lines, where the bottom (= red) detour is chosen.

Region 5. The same discussion applies to the case when p is located in Regions 3, 7, or 9. (3) When p is located in Region 5, it is not connectable to any region.

Given the merging segments of the parent p and its child u, the *Nine-Region-Based Cutting* method consists of two steps. First, it constructs nine regions for a routing obstacle. Correspondingly, a merging segment is divided into several sub-segments, where each sub-segment belongs to a unique region. Second, it checks each sub-segment of the parent p to see if it is connectable to any sub-segment of the child u. In addition, the distance between these two sub-segments should be equal to the shortest distance between $ms(p)$ and $ms(u)$. If these conditions are satisfied, the current sub-segment of the parent is routing-overlap free. Figure 6.b) shows a sample, where $(p1, p2)$ and $(p2, p3)$ are the FMS of $ms(p)$. Note that $(p3, p4)$ is not a FMS since the distance between $(p3, p4)$ and $(u2, u3)$ is longer than the shortest distance between $ms(p)$ and $ms(u)$. This technique also helps us determine the actual routing topology during the top-down embedding procedure, when a routing obstacle presents in the merging region.

VI. TSV-Obstacle-Aware Detouring

In this section, we discuss two major cases when no feasible merging segment exists within the merging region of the shortest distance: one is for routing obstacles, the other is for placement obstacles. We develop two detour policies to find the feasible merging segments outside the merging region.

A. Routing-Obstacle-Aware Detour

When a routing obstacle blocks the routing region, we use the routing-obstacle-aware detour technique to find the feasible merging segments outside the merging region. This situation is usually caused by big P/G TSVs. In this case, the merging segment of the parent is not connectable to that of its children $ms(u)$ and $ms(v)$. Figure 7 shows two detour cases, when the merging segment is a point (Figure 7.a) or an arc (Figure 7.b).

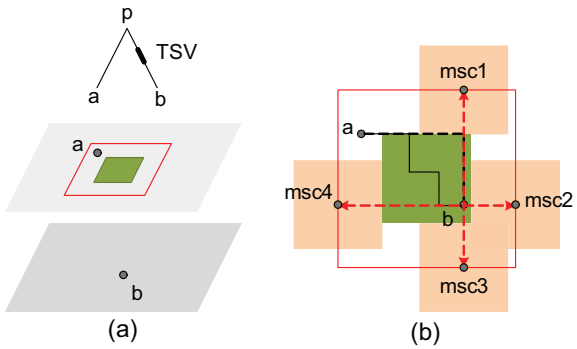

Fig. 8. Placement-obstacle-aware detour for TSV merging. A signal TSV occupies the merging area between nodes a and b where a TSV is needed. A feasible merging segment for this clock TSV is added on the expanded-overlap-free boundary with the shortest merging distance. $msc1$-$msc4$ show four candidates. We choose $msc2$ due to its shortest distance to b.

A new merging point location of the parent p' is chosen along the boundary of the obstacle that the u-to-p'-to-v wirelength (L') is minimized. We then calculate the merging distance between nodes p', u and p', v based on the zero-skew equations. In this case, L' becomes the shortest distance to connect $ms(u)$ and $ms(v)$ while traveling along the obstacle boundary. And p' is the zero-skew point in between u and v.

B. Placement-Obstacle-Aware Detour

We use a TSV merging example to show how our detour policy works in the case that no feasible merging segment exists when inserting a clock TSV. When merging a node a in the top die and b in the bottom die at their merging segment located on the top, the expanded-overlap-free boundary of the signal TSV (= placement obstacle) may cover both nodes a and b. As a result, the placement-overlap-free merging segment of the clock TSV to be added exists in the merging region. As shown in Figure 8.a), b in the bottom die is allowed to have overlap with the signal TSVs in the top die. Referring to the top-down view in Figure 8.b), our detour policy is to extend node b in four directions, and obtain intersections along the expanded-overlap-free boundary of the obstacle, *i.e.*, $msc1$ to $msc4$, which are the four potential locations of the merging segments of the clock TSV. We choose $msc2$, which is the nearest intersection to b, update the distance between $msc2$ and b, and the merging distance between a and $msc2$. By solving the zero-skew equations, we can then determine the merging segment of the parent p.

VII. Clock TSV Merging

We observe that a longer feasible merging segment helps avoid TSV-induced overlap with clock buffers and other TSVs. Our policy is to find the longest feasible merging segment by sweeping the distance between the clock TSV and its child. Figure 9 shows an illustration on clock TSV merging.

We first determine the merging segment for the clock TSV, denoted $msc(TSV)$, using $d3$, which is the distance between $msc(TSV)$ and $ms(v)$, the merging segment of the child v. We then apply the *Nine-Region-Based Cutting* and *Expanded-Obstacle Cutting* methods to decide the feasible merging segment for the clock TSV. After that, we determine the merging segment of the parent p, denoted $msp(p)$, by deriving the distances $d1$ and $d2$ based on the conventional zero-skew constraint and the shortest merging distance requirement. The

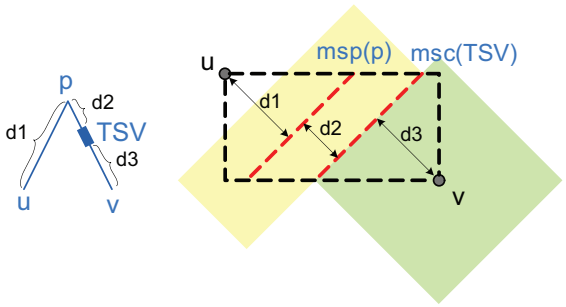

Fig. 9. Finding the longest feasible merging segment for the clock TSV by sweeping the distance between clock TSV and $ms(v)$.

TABLE I
BENCHMARK INFORMATION. FOOTPRINT AREA IS IN μm^2.

Ckt	Area	#Sinks (die-0 + die-1)	#Signal TSVs	#P/G TSVs
IDCT	433^2	117 + 356	342	82
8086	420^2	230 + 427	323	82
8051	400^2	347 + 1009	306	64
b18	483^2	1652 + 1448	440	100
b19	590^2	3099 + 3071	462	144

feasible merging segment for p is determined using the *Nine-Region-Based* Cutting method in this case.

To find the longest $msc(TSV)$, we sweep the distance $d3$ with a certain step (such as $d/2$). Under a given $d3$, we obtain a pair of the feasible merging segments for the TSV and p. We choose the longest length of the feasible merging segment for the TSV as our final merging solution. This scheme is shown to provide a better chance to avoid overlap between clock buffers and clock TSVs during the top-down embedding.

VIII. EXPERIMENTAL RESULTS

A. Simulation Setting

We apply our TSV-obstacle-aware clock routing method to the IWLS 2005 benchmarks [12], as listed in Table I. We use 45nm technology. The P/G TSV cell is $12.35\mu m \times 12.35\mu m$, the signal and clock TSV cells are $7.41\mu m \times 7.41\mu m$. And the clock buffer cell occupies $2.09\mu m \times 2.47\mu m$.

For each benchmark, we perform 3D power planning and gate/TSV placement using a Cadence Encounter-based 3D physical design tool-chain. We obtain a TSV obstacle map (including P/G and signal TSVs) and clock sinks locations. Table I shows the benchmark information. We then apply our TSV-obstacle-aware 3D clock routing algorithm to achieve an overlap-free 3D clock tree under the given maximum slew rate, TSV count bound, and zero-skew (under the Elmore delay model) constraints. Then we apply SPICE simulation on entire 3D clock network to report clock power consumption and timing. The clock frequency is set to $1GHz$, with the supply voltage $1.1V$. The maximum clock skew from the simulation is required to be under $30ps$. The maximum loading capacitance for each clock buffer is $100fF$. TSV capacitance is $15fF$, and resistance is $35m\Omega$. The clock source is located on the topmost die (= die-0).

B. Sample TSV-Aware Clock Topology

Figures 10 and 11 show 3D clock routing result of benchmark $b19$, where the first ignores TSV obstacles (= Figure 10) and the second avoids TSV obstacles (= Figure 11). Both results are based on the same set of two-die stack clock sinks, TSV obstacle map, and clock constraints. We highlight several dense regions in the tree and show the details inside. We observe that many violations (= illegal overlaps) occur, especially in the dense regions, including clock TSV overlap with other P/G and signal TSVs, routing over P/G TSVs, buffer and signal TSV overlap. However, by using our TSV-obstacle-aware clock routing algorithm, we see that no clock net is routed over the P/G TSVs, and no overlap exists among P/G TSVs, signal TSVs, clock TSVs, and clock buffers.

C. Impact of TSV-Induced Obstacles

We compare the quality of the clock trees with and without TSV obstacle avoidance to quantify various kinds of overhead that occur by avoiding TSV obstacles. Table II shows a comparison of wirelength, clock skew, clock slew and clock power under the same amount of clock TSVs and the same clock buffer used. We also show the % increase of wirelength and power of the TSV-obstacle-aware clock routing results over the obstacle-ignoring cases.

First, our TSV obstacle-aware clock routing algorithm is able to achieve a TSV-overlap-free clock tree. Second, the clock skews are all zero under the Elmore delay model, and are well controlled under $30ps$ from SPICE simulation. Third, our TSV-obstacle-aware clock routing results are comparable to the result when TSV obstacles are ignored. We show two cases of clock TSV usage, one that uses a small number of TSVs, the other one that uses a larger number of TSVs. In most of the cases, the TSV-obstacle-aware clock tree has slightly larger wirelength or clock power; in some benchmarks, TSV-obstacle-aware clock routing obtain slightly better results. This demonstrates that our TSV obstacle avoidance method works well while keeping the overhead almost negligible. Moreover, the runtime of our TSV-obstacle-aware clock routing is within several seconds.

IX. CONCLUSIONS

In this paper, we addressed a practical obstacle issue in TSV-based 3D clock tree synthesis and studied how to avoid TSV-induced obstacles in 3D clock routing. We first discuss how power/ground TSVs (P/G TSVs) and signal TSVs become two different types of obstacles in 3D clock routing. We then developed a TSV-obstacle-aware clock routing algorithm to construct a TSV-overlap-free buffered clock tree. We proposed a TSV-obstacle-aware DME technique. We also studied how to apply detour when no feasible merging segment exists. Experiments show that we can achieve a buffered clock tree that avoids TSV-induced obstacles while keeping the wirelength and power overhead to a minimum.

REFERENCES

[1] J. Minz, *et al.*, "Buffered Clock Tree Synthesis for 3D ICs Under Thermal Variations," in *Proc. Asia and South Pacific Design Automation Conf.*, 2008, pp. 504–509.
[2] X. Zhao, *et al.*, "Low-Power and Reliable Clock Network Design for Through-Silicon via (TSV) Based 3D ICs," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 1, no. 2, pp. 247–259, 2011.
[3] T.-Y. Kim and T. Kim, "Clock Tree Embedding for 3D ICs," in *Proc. Asia and South Pacific Design Automation Conf.*, 2010, pp. 486–491.
[4] X. Zhao, *et al.*, "Pre-bond Testable Low-Power Clock Tree Design for 3D Stacked ICs," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, 2009, pp. 184–190.
[5] T.-Y. Kim and T. Kim, "Clock Tree Synthesis with Pre-Bond Testability for 3D Stacked IC Designs," in *Proc. ACM Design Automation Conf.*, 2010, pp. 723–728.
[6] A. B. Kahng and C.-W. A. Tsao, "More Practical Bounded-Skew Clock Routing," in *Proc. ACM Design Automation Conf.*, 1997, pp. 594–599.
[7] H. Kim and D. Zhou, "Efficient Implementation of a Planar Clock Routing with the Treatment of Obstacles," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 10, pp. 1220–1225, 2000.
[8] H. Huang, *et al.*, "DME-Based Clock Routing in the Presence of Obstacles," in *in Proceedings of 7th International Conference on ASIC*, 2008, pp. 429–434.

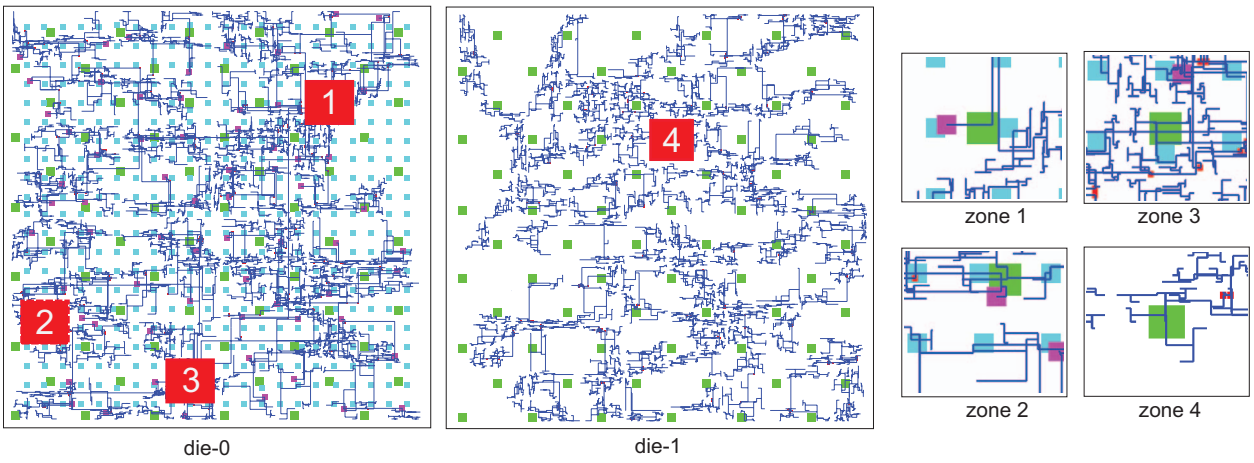

Fig. 10. A two-die stack clock routing WITHOUT considering TSV obstacles. We show P/G TSVs (green), signal TSVs (blue), clock TSVs (red), clock wires, and clock buffers (red). This tree violates several overlapping constraints, including clock TSVs overlap with other P/G TSVs, signal TSVs, and buffers, and routing over P/G TSVs.

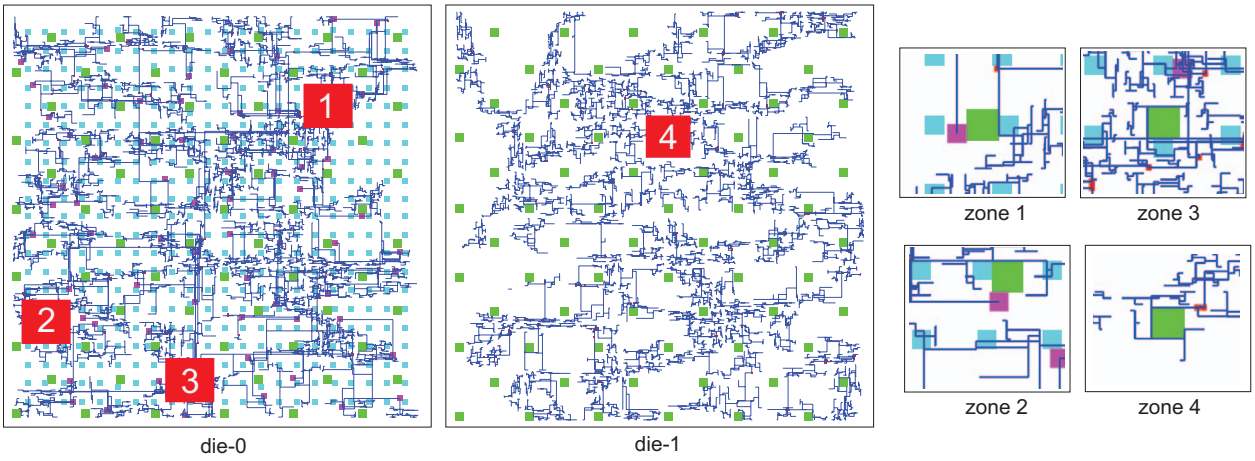

Fig. 11. A two-die stack clock tree WITH TSV obstacle avoidance for the same circuit as Fig 10. This tree does not contain any illegal overlap.

TABLE II

COMPARISON OF TWO 3D CLOCK ROUTING RESULTS. THE FIRST ONE AVOIDS TSV OBSTACLES BY APPLYING TSV-OBSTACLE-AWARE ROUTING; AND THE SECOND ONE IGNORES TSV OBSTACLES. WE ALSO SHOW % INCREASE OF CLOCK POWER AND WIRELENGTH OF TSV-OBSTACLE-AWARE ROUTING.

		Avoid TSV obstacles				Ignore TSV obstacles				Increase (%)	
ckt	#TSVs	WL (um)	#Bufs	Pwr (mW)	Skew (ps)	WL (um)	#Bufs	Pwr (mW)	Skew (ps)	WL (um)	Pwr (mW)
Using small amount of clock TSVs											
IDCT	4	21431	178	16.6	17.7	21810	178	16.7	19.7	-1.7	-0.3
8086	4	25055	131	12.3	19.3	25223	125	12.0	22.4	-0.7	3.1
8051	5	50128	456	41.5	15.0	47616	449	40.8	15.4	5.3	1.6
b18	10	75653	471	42.4	23.9	74964	468	42.1	20.4	0.9	0.9
b19	10	160621	823	81.8	16.4	158082	818	81.0	15.7	1.6	1.0
Using more clock TSVs											
IDCT	10	22069	172	16.7	19.7	22146	170	16.5	18.5	-0.3	0.9
8086	14	22475	110	10.7	21.1	22459	110	10.7	21.1	0.1	0.0
8051	35	51627	439	42.2	11.7	52007	438	42.2	11.6	-0.7	-0.1
b18	58	91273	517	50.7	14.2	90811	515	50.5	14.3	0.5	0.4
b19	70	146603	792	78.2	20.9	145798	791	78.0	20.7	0.6	0.2

[9] W.-H. Liu, *et al.*, "Minimizing Clock Latency Range in Robust Clock Tree Synthesis," in *Proc. Asia and South Pacific Design Automation Conf.*, 2010, pp. 389–394.

[10] J. Lu, *et al.*, "A Dual-MST Approach for Clock Network Synthesis," in *Proc. Asia and South Pacific Design Automation Conf.*, 2010, pp. 467–473.

[11] X.-W. Shih, *et al.*, "Blockage-Avoiding Buffered Clock-Tree Synthesis for Clock Latency-Range and Skew Minimization," in *Proc. Asia and South Pacific Design Automation Conf.*, 2010, pp. 395–400.

[12] IWLS2005 Benchmark. http://www.iwls.org/iwls2005/.