Low-Power and Reliable Clock Network Design for Through-Silicon Via (TSV) Based 3D ICs

Xin Zhao, *Student Member, IEEE*, Jacob Minz, *Member, IEEE*, and Sung Kyu Lim, *Senior Member, IEEE*

***Abstract*—This paper focuses on low-power and low-slew clock network design and analysis for through-silicon via (TSV) based three-dimensional stacked ICs (3D ICs). First, we investigate the impact of the TSV count and the TSV resistance–capacitance (*RC*) parasitics on clock power consumption. Several techniques are introduced to reduce the clock power consumption and slew of the 3D clock distribution network. We analyze how these design factors affect the overall wire length, clock power, slew, and skew in 3D clock network design. Second, we develop a two-step 3D clock tree synthesis method: 1) 3D abstract tree generation based on the three-dimensional method of means and medians (3D-MMM) algorithm; 2) buffering and embedding based on the slew-aware deferred-merge buffering and embedding (sDMBE) algorithm. We also extend the 3D-MMM method (3D-MMM-ext) to determine the optimal number of TSVs to be used in the 3D clock tree so that the overall power consumption is minimized. Related SPICE simulation indicates that: 1) a 3D clock network that uses multiple TSVs significantly reduces the clock power compared with the single-TSV case, 2) as the TSV capacitance increases, the power savings of a multiple-TSV clock network decreases, and 3) our 3D-MMM-ext method finds a close-to-optimal design point in the “TSV count versus power consumption” tradeoff curve very efficiently.**

***Index Terms*—3D clock network, clock slew, low-power 3D IC design, through-silicon via (TSV).**

I. Introduction

IN three-dimensional integrated circuits (3D ICs), the clock distribution network spreads over the entire stack to distribute the clock signal to all the sequential elements. Clock skew, defined as the maximum difference in the clock signal arrival times from the clock source to all sinks, is required to be less than 3% or 4% of the clock period in an aggressive clock network design according to the International Technology Roadmap for Semiconductors (ITRS) projection [1]. Thus, clock skew control, which was well studied in 2D ICs [2], is still a primary objective in the 3D clock network design. However, the clock signal in 3D ICs is distributed not only along the X and Y directions, but also along the Z direction using through-silicon vias (TSVs). The clock distribution network drives large capacitive loads and switches at a high frequency. This leads to an increasingly large proportion of the total power dissipated in the clock distribution network. In some applications, the clock network itself is responsible for 25% [3] and even up to 50% [4] of the total chip power consumption. Moreover, clock slew must also be taken into consideration when designing a 3D clock network, because a large clock slew may cause a setup/hold time violation. Thus, low power, skew, and slew remain important design goals in 3D clock networks.

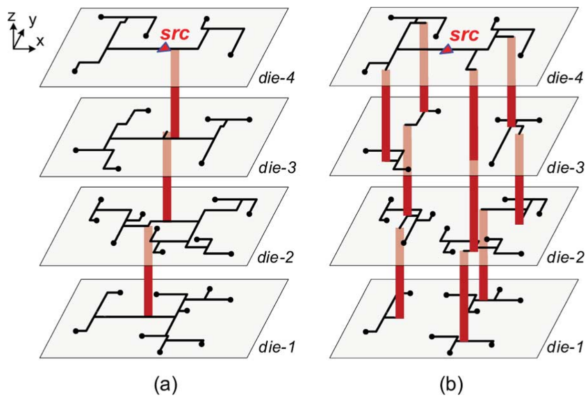

Fig. 1. Four-die stack 3D clock networks with two different TSV counts. (a) Single TSV between adjacent dies. (b) Ten TSVs. The overall wire length is shorter in (b).

TSVS provide the vertical interconnections to deliver the clock signal to all dies in the 3D stack. The TSV count is an important factor that characterizes the physical and electrical properties of the clock network. 3D integration with TSVs has been intensively studied in both chip-to-chip and chip-to-wafer communications [5]. The fabrication and characterization of TSVs are being explored in many companies and institutions [6]. TSV reliability issues are also studied [7].

The low-power 3D clock network design demands a thorough investigation on how the TSV count and TSV parasitics affect the clock performance. Existing work has demonstrated that the total wire length of a 3D clock network decreases significantly if more TSVs are used [8]–[11]. According to the observations made in [8], the die that contains the clock source includes a complete tree, while other dies can have subtrees, as illustrated in Fig. 1. A 3D clock tree that utilizes multiple TSVs tends to reduce the overall wire length as more and more TSVs are used. However, the analysis of TSV resistance–capacitance (*RC*) parasitics on the clock network has not been addressed in the literature. If a 3D clock tree utilizes many TSVs that have large TSV *RC* parasitics, the clock delay and power consumption contributed by the TSVs may increase significantly. Using

Manuscript received June 14, 2010; revised October 04, 2010; accepted November 09, 2010. Date of publication December 23, 2010; date of current version March 23, 2011. This work was supported in part by a grant from SRC Interconnect Focus Center (IFC) and in part by the National Science Foundation (NSF) CAREER Grant CCF-0546382. This work was recommended for publication by Associate Editor D. G. Kam upon evaluation of reviewers comments.

X. Zhao and S. K. Lim are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA.

J. Minz is with the Synopsys Corporation, 560016 Bangalore, India.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCPMT.2010.2099590

more TSVs helps to reduce the wire length and thus power consumption, but the TSV capacitance increases the clock power consumed at the same time. Our experiments indicate that the larger the TSV capacitance is, the faster the clock power consumption increases when more and more TSVs are used.

In this paper, we investigate the impact of various design parameters on the wire length, clock power, slew, and skew of the 3D clock network. These parameters include the TSV count[1], TSV parasitics, the maximum loading capacitance of the clock buffers, and the supply voltage. We also develop clock network synthesis algorithms for low-power 3D clock network design. The contributions of this paper are as follows.

- We provide an extensive study on the impact of the TSV count and TSV parasitics on the clock power. We show the "TSV count versus power dissipation" tradeoff curves for various TSV parasitic values and discuss how the TSV count and the TSV capacitance together determine the overall clock power consumption.
- We discuss the impact of the TSV count and clock buffer insertion on clock slew control. Our study shows that using multiple TSVs helps to reduce the maximum and average slew as compared with the single-TSV case. In addition, specifying an upper bound of the load capacitance for each clock buffer remains an efficient way to control the maximum slew of the 3D clock network design.
- We present an effective way to determine the optimal number of TSVs for the 3D clock tree so that the overall power consumption is minimized. Our method predicts the impact of adding a new TSV into the current clock topology on the overall power consumption during the top-down abstract tree generation. This helps to decide whether pairing of two clock nodes in different dies and using a TSV for this pair is useful for power reduction or not. Related experiments indicate that our method finds a close-to-optimal design point in the TSV count versus power consumption tradeoff efficiently as compared with a straightforward exhaustive search method.

The organization of this paper is as follows. We present an overview of related work on 3D clock network design in Section II. We formulate the problem of 3D clock tree synthesis in Section III. Section IV presents our 3D clock routing algorithm. We present an extension of our 3D-MMM algorithm in Section V. Section VI presents experimental results. We conclude the paper in Section VII.

II. Related Work

In 3D clock tree design and optimization, TSV planning plays an important role in constructing a low-power 3D clock network. Minz *et al.* [8] proposed the first work on 3D clock routing algorithms. They discovered that the total wire length decreases significantly when more TSVs are used in the 3D clock network. They also studied the thermal impact on the 3D clock network, and proposed a thermal-aware 3D clock tree synthesis method to balance the clock skew caused by the thermal variations. Kim and Kim [11] presented a 3D embedding method to reduce wire length. However, they do not consider power consumption or slew rate and do not provide any SPICE simulation results. Zhao *et al.* [9] developed a clock design method to support the pre-bond testing for 3D ICs. They also discussed the impact of the TSV counts on the pre-bond testable clock tree. They observed that using multiple TSVs helps a pre-bond testable 3D clock network achieve low power consumption. However, this work did not take into account the impact of the TSV capacitance on the clock power.

Pavlidis *et al.* [12] presented measurement data on a fabricated 3D clock distribution network. Arunachalam and Burleson [13] proposed the use of a separate layer for the clock distribution network to reduce power. Their simulations show 15%–20% power reduction over the same 2D chip clock network. However, they focused on a simple H-tree and did not perform any design-level optimization.

Due to the significant dimension of the TSVs occupying the layout space [14], the impact of TSV parasitics, especially the capacitance, should be taken into consideration for a low-power 3D clock network design. Existing work mainly focuses on 15 fF TSVs. But the TSV capacitance can vary from tens of femtofarads up to a few hundred femto-farads, depending on the material, TSV diameter, oxide thickness, and TSV height [15], [16]. We observe that large TSV capacitance values significantly affect the existing discussions on the TSV count versus power tradeoff. In this case, the multiple-TSV insertion reduces the total wire length and power, but the large TSV parasitic capacitance increases the power consumption. As a result, the total clock power may increase in the multiple-TSV case. Therefore, a thorough study on the impact of both the TSV capacitance and the TSV count on the overall 3D clock power is required. Given the TSV parasitic impedance, a straightforward approach to find the optimal TSV usage for low-power design is to exhaustively search the entire range of the TSV count. This approach, however, requires prohibitive design time and is thus not practical.

III. Preliminaries

A. Electrical and Physical Model of 3D Clock Network

In this paper, a 3D clock network is modeled as a distributed *RC* network. The sink nodes that represent flip-flops, and clock input pins of IP or memory blocks, are modeled as capacitive loads. Wire segments and TSVs are modeled with a π model[2], which is a classical way to represent the parasitics of a clock network. Each buffer or driver is constructed with two inverters. Note that prior work has focused on the electrical modeling of TSVs [15]–[18]. Our 3D clock routing algorithm is flexible to handle a more complicated TSV parasitic model if desired.

The *TSV bound* is defined as a user specified constraint on the maximum allowed TSV number per die. The TSV bound is usually decided before clock synthesis and is based upon the process technology. Different from the TSV bound, the *TSV count (#TSVs)* is the total number of TSVs utilized in the 3D

[1]In this paper, we use "TSV count" to refer to the total number of TSVs used in a 3D clock tree.

[2]In this work, wire segments denote the edges of the abstract tree, and are not uniformly distributed. Depending on the TSV insertion and buffer insertion on the abstract tree, a src-to-sink path usually contains tens of wire segments, with each segment length varies from tens of micro-meters to a few hundreds of micro-meters.

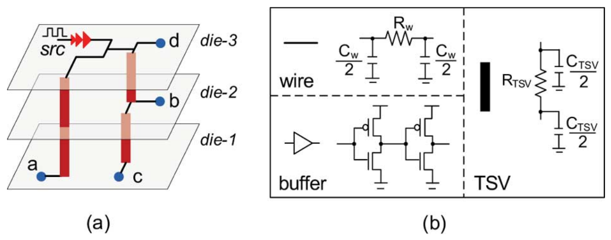

Fig. 2. A sample clock tree and its electrical model. (a) A sample three-die clock network using four TSVs, where the clock source is in die-3. Sink a in die-1 uses two TSVs that are vertically aligned, and sink b in die-2 uses one TSV, to connect to the clock source. (b) Electrical models of the clock wire segments, TSVs, and buffers/drivers.

clock tree. For an n-die 3D stack, #TSVs is usually less than or equal to $(n-1)\times \mathrm{TSV}$ bound.

A three-die clock interconnect using four TSVs is shown in Fig. 2: the clock source is located in die-3; sink a in die-1 connects to the source using two TSVs that are vertically aligned; sink c in die-1 connects to the source by two TSVs; and sink b in die-2 uses one TSV.

B. Problem Formulation

The *3D Clock Tree Synthesis* problem can be formulated as following: Given a set of sinks in all dies, a TSV bound, a predetermined clock source location, and the parasitics of the wires, buffers, and TSVs, the 3D clock tree synthesis constructs a fully-connected 3D clock network such that 1) clock sinks in all dies are connected by a single tree, 2) the TSV count in each die is under the TSV bound, 3) the clock skew is minimized (and zero under the Elmore delay model [19]), 4) clock slew is below the constraint, and 5) the wire length and clock power are minimized.

The clock skew is the maximum difference among the arrival times at the clock sinks. In the existing clock tree synthesis tools, the Elmore delay model is a popular measure of the *RC* delay and skew. The primary goal of our 3D clock tree synthesis is to guarantee a zero Elmore-skew clock network. In order to achieve more accurate timing information and to evaluate our clock synthesis performance, we use SPICE simulation on our 3D clock trees. The simulated clock skew is constrained to less than 3% of the clock period. The clock slew is defined as the transition time from 10% to 90% of the clock signal at each sink.

The TSV bound constraint plays an important role in achieving low-power 3D clock networks. It reflects the impact of TSV usage on routing congestion, capacitive coupling, stress-induced manufacturing issues, and so on. By varying the TSV bounds, we obtain different 3D clock networks with different qualities. Note that the TSV bound and the actual TSV usage in each die may be different, because the bound only puts the limit on the maximum TSV usage for each die.

IV. 3D Clock Tree Synthesis

A. Overview

Our 3D clock tree synthesis algorithm consists of two major steps: 1) 3D abstract tree generation and 2) slew-aware buffering and embedding. First, we generate a 3D abstract tree based on our 3D method of means and medians (3D-MMM) algorithm. The 3D-MMM algorithm basically determines which pair of nodes (sink nodes or merging points) to connect together and utilizes TSVs if necessary, while building a binary tree in a top-down fashion. Note that our 3D-MMM algorithm works in such a way that there is always one die that contains a *single tree* which connects all sinks in the die, whereas the sinks in other dies are connected with *multiple trees*. In this case, the clock source is located in the die that contains the single tree.

Once a 3D abstract tree is obtained, we determine the routing topology and exact geometric locations for all the nodes, TSVs, and buffers. Our *slew-aware deferred-merge buffering and embedding* (sDMBE) method is a two-phase approach, which is based on the classic deferred-merge and embedding (DME) algorithm [20] in 2D clock routing. sDMBE first visits each node in a bottom-up fashion, determines the merging type for a pair of subtrees, inserts buffers if necessary, and calculates the merging distances based on the zero-Elmore-skew equations. The outcomes of sDMBE during the first phase are the merging segments, which store a collection of feasible locations of the internal nodes in the 3D abstract tree. During the second phase, sDMBE visits the whole abstract tree in a top-down manner while deciding the exact merging locations of the internal nodes, buffers, TSVs, and exact routing topology until all sinks are connected via a single tree.

B. 3D Abstract Tree Generation

The first step of our 3D clock tree synthesis is the 3D abstract tree generation using the 3D-MMM algorithm. A 3D abstract tree indicates the hierarchical connection information among the sink nodes, internal nodes, TSVs, and the root node. The 3D abstract tree of an n-die stack clock network is an n-colored binary tree, which is used to identify the die index for all the nodes.

We develop the 3D-MMM algorithm to generate a 3D abstract tree for the given clock sinks in a top-down manner, which is an extension of the method of means and medians (MMM) algorithm [21]. The 3D abstract trees generated by the 3D-MMM algorithm with various TSV bounds are shown in Fig. 3. Note that a larger TSV bound tends to move TSVs closer to the sink nodes and causes more vertical clock connections than horizontal connections. However, the overall wire length is reduced due to the short horizontal connection length. The basic idea of our 3D-MMM algorithm is to recursively divide the given sink set into two subsets until each sink belongs to its own set. A TSV is used if we decide to merge a pair of nodes in different dies. In this case, our goal is to evenly distribute the TSVs across the die area and to satisfy the given TSV bound, which is shown to improve manufacturability [22].

Let $S=\{s_1,s_2,\ldots,s_k\}$ denote a set of sinks, where the locations of the sinks have been decided before the 3D clock tree synthesis. We assume that the maximum allowed TSV count for each die in S (TSV bound) is also given. Each s_i is a triplet of (x_i,y_i,z_i), where z_i is the die index of s_i, and x_i and y_i are the X and Y coordinates of s_i. Let $stack(S)$ denote the number of dies the sinks in set S are located in. In each recursive partitioning, we divide the set S into two subsets S_1 and S_2 based on the following two cases.

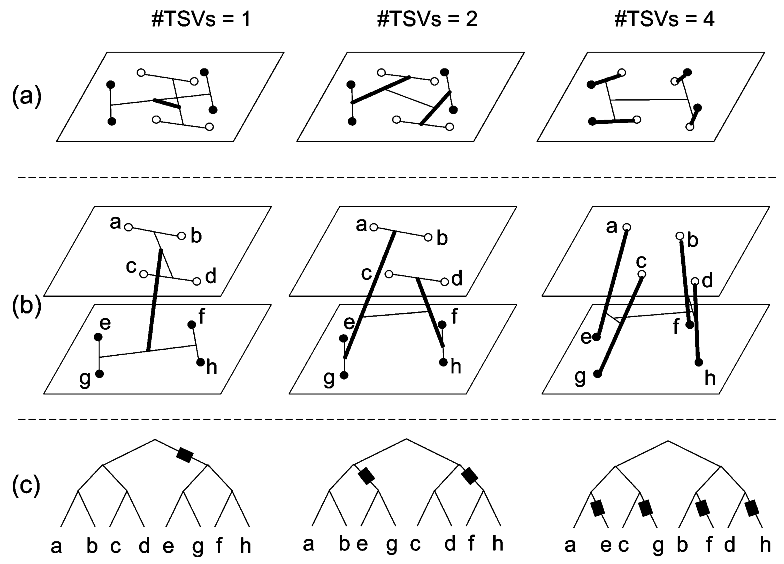

Fig. 3. The 3D abstract trees generated by our 3D-MMM algorithm under various TSV bounds. (a) 2D view where thick lines denote TSV connection. (b) 3D view. (c) Binary abstract trees where the squares denote TSVs.

```
Z-cut(SinkSet $S$, Subset $S_T$, Subset $S_B$)
Input: Sink set $S = \{s_1, \cdots, s_k\}$, source die index $Z_s$
Output: Subsets $S_T$ and $S_B$

1: $Z_{\min} = min(z_1, .., z_i, .., z_k)$, $s_i = (x_i, y_i, z_i) \in S$
2: $Z_{\max} = max(z_1, .., z_i, .., z_k)$, $s_i = (x_i, y_i, z_i) \in S$
3: if ($Z_s \leq Z_{min}$) then
4:     $S_T = \{s_1, .., s_i, .., s_{k1}\}$, $z_i \in [Z_{\min}+1, Z_{\max}]$
5:     $S_B = \{s_{k1+1}, .., s_j, .., s_k\}$, $z_j = Z_{\min}$
6: else if ($Z_s \geq Z_{\max}$) then
7:     $S_T = \{s_1, .., s_i, .., s_{k1}\}$, $z_i = Z_{\max}$
8:     $S_B = \{s_{k1+1}, .., s_j, .., s_k\}$, $z_j \in [Z_{\min}, Z_{\max}-1]$
9: else
10:    $S_T = \{s_1, .., s_i, .., s_{k1}\}$, $z_i = Z_s$
11:    $S_B = \{s_{k1+1}, .., s_j, .., s_k\}$, $z_j \neq Z_s$
```

Fig. 4. Pseudo code of the Z-cut procedure, which corresponds to the line 6 in the 3D-MMM algorithm, Fig. 5.

```
3D Abstract Tree Generation (3D-MMM)
Input: clock sinks in 3D and a TSV bound
Output: a rooted 3D abstract tree

1: AbsTreeGen3D(SinkSet $S$, bound $B$)
2:     $S_1$ and $S_2$ = subsets of $S$;
3:     if ($|S|$ = 1) then
4:         return $root(S)$;
5:     else if (B = 1 and $stack(S) > 1$) then
6:         Z-cut($S$, $S_1$, $S_2$);
7:         $B_1$ = $B_2$ = 1;
8:     else
9:         Geometrically divide $S$ into $S_1$, $S_2$;
10:        Find $B_1$, $B_2$ such that $B_1$ + $B_2$ = $B$;
11:    $root(S_1)$ = AbsTreeGen3D($S_1$, $B_1$);
12:    $root(S_2)$ = AbsTreeGen3D($S_2$, $B_2$);
13:    $leftChild(root(S))$ = $root(S_1)$;
14:    $rightChild(root(S))$ = $root(S_2)$;
15:    return $root(S)$;
```

Fig. 5. Pseudo code of the 3D-MMM algorithm.

- Z-cut: if the TSV bound is one, the given sink set S is partitioned such that the sinks from the same die belong to the same subset. The connection between S_1 and S_2 needs one TSV in-between adjacent dies. Note that 3D-MMM is a bi-partitioning process. When the sinks of S distribute to more than two dies (i.e., $stack(S) > 2$), we need $stack(S) - 1$ iterations of Z-direction partitions to split the sink set into subsets, so that the sinks belonging to the same die are in the same subset. Furthermore, the order of the Z-cut also depends on the source die index. Fig. 4 shows the details of the Z-cut procedure.
- X/Y-cut: if the TSV bound is larger than one, or the sinks in the set S belong to the same die, the set S is partitioned geometrically by a horizontal line (X-cut or Y-cut), and Z-dimension is ignored. If the subsets contain sinks from different dies, we potentially need multiple TSVs to connect those sinks.

At the end of each partitioning, we propagate the TSV bound constraint by assigning a TSV bound for each new subset.

The 3D abstract tree generation using the 3D-MMM algorithm is shown in Fig. 5. The recursive method takes as inputs a set of 3D clock sinks and a TSV bound. If the size of the given sink set (i.e., $|S|$) is one, then we reach the bottom level of the abstract tree (lines 3–4). If the TSV bound is one, Z-cut is applied to partition the sink set S into two subsets S_1 and S_2 (lines 6–7). As previously discussed, once the TSV bound is one, our 3D-MMM performs $stack(S)-1$ Z-direction partitions, so that the sinks belong to the same die are in the same subset. In order to guarantee that only one TSV is used between adjacent dies, the order of die-wise Z-cut depends on the source-die index and the die indices in the sink set S, as illustrated in Fig. 4. In the case that the above conditions are not satisfied, the set S is partitioned geometrically by a horizontal line (X-cut or Y-cut), so called X/Y-cut (line 9). And the Z-dimension of each sink is ignored. The cut line is drawn at the median of the X or Y coordinates of the sinks. The TSV bound is divided for the two subsets (line 10). The bound for each subset is calculated by

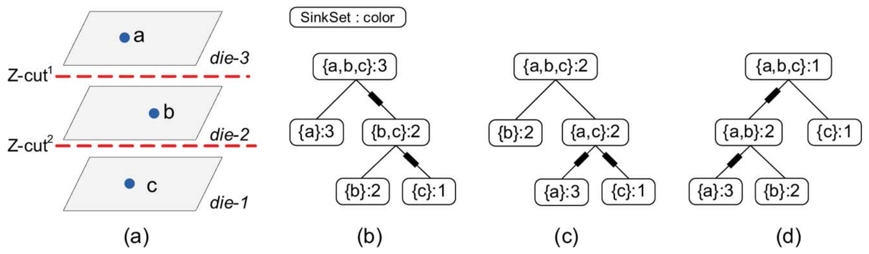

Fig. 6. Three-colored 3D abstract trees after applying Z-cut twice on the three-die stack sink set $\{a, b, c\}$, when the clock source is located in (b) die-3, (c) die-2, and (d) die-1. Each node in the abstract tree contains the corresponding sink set and a color index (b) first applies Z-cut^1 and then Z-cut^2, whereas (d) applies Z-cut^2 first and then Z-cut^1.

1) estimating the number of TSVs required by each subset and 2) dividing the given bound B according to the ratio of the estimated TSVs. For each subset, we assume the minimum sink size in each die as the estimation of the number of TSVs. The procedure is called recursively for each of the subsets S_1 and S_2 with different TSV bounds (lines 11–12). The roots of the subtrees are connected by the root of the higher-level tree (lines 13–15). The complexity of the algorithm is $O(n \cdot logn)$, where n is the number of nodes.

Corresponding to the n-die stack clock sinks, the 3D abstract tree is an n-colored binary tree, where each node (i.e., sinks, internal nodes, and the root) is assigned a color to represent which die it belongs to. The dies are numbered from 1 to n from the bottom to the top. Let $c(p)$ be the color index for node p, where $c(p) \in \{1, 2, \ldots, n\}$. For example, $c(p) = 1$ means that node p is located in die-1. Let $c(src)$ denote the die index, where the clock source is located. During the top-down 3D abstract tree generation, we color the nodes corresponding to the sink sets. Considering the node p with the sink set S, let $Z_{\max}$ and $Z_{\min}$ be the maximum and minimum die indices of the sinks within set S. The color of p is determined as follows:

$$c(p) = \begin{cases} c(src), & \text{if } p \text{ is the root} \\ Z_{\min}, & \text{else if } Z_{\min} > c(src) \\ Z_{\max}, & \text{else if } Z_{\max} < c(src) \\ c(src), & \text{otherwise.} \end{cases} \quad (1)$$

Considering an edge e with two terminal nodes n_1 and n_2. The following are true: 1) if $c(n_1) = c(n_2)$, the edge e will be routed in the same die as node n_1 and n_2; 2) if $c(n_1) \neq c(n_2)$, then $|c(n_1) - c(n_2)|$ number of TSVs will be inserted along the edge e. Fig. 6 shows an illustration, where 3D abstract trees for a sink set $\{a, b, c\}$ are shown after applying Z-cut twice. Fig. 6(b)–(d) are the three abstract trees, where the clock source is located in die-3, die-2, and die-1, respectively. Each node in the abstract tree contains the sink set and color information. The abstract trees in Fig. 6(b) is obtained by Z-cut^1 first and then Z-cut^2. Whereas, Fig. 6(d) applies Z-cut^2 first and then Z-cut^1. Fig. 6(c) first extracts the sinks of the clock source die, and then applies a Z-cut. The primary goal of using a different Z-cut sequence is to guarantee that only one TSV is necessary between adjacent dies after $stack(S) - 1$ Z-cuts.

C. Slew-Aware Buffering and Embedding

The second step of our 3D clock tree synthesis is slew-aware buffering and embedding: Given a 3D abstract tree, the goal is to determine the exact geometric locations of all the nodes, TSVs, and buffers, such that the wire length of the embedded and buffered clock tree is minimized, the load capacitance of each buffer does not exceed the predefined maximum value (CMAX), and the clock skew is zero under the Elmore delay model. We develop the slew-aware deferred-merge buffering and embedding (sDMBE) algorithm to geometrically embed (route) the abstract tree.

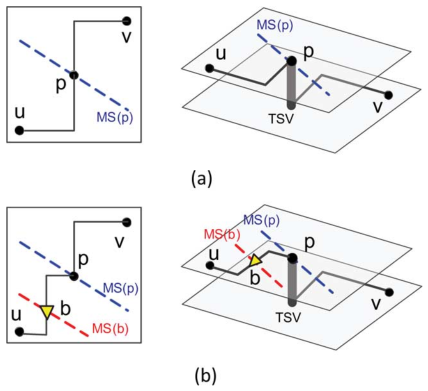

Fig. 7. Samples of 3D merging segments for (a) an unbuffered tree, and (b) a buffered tree.

sDMBE is a two-phase algorithm and is based on the deferred-merge embedding (DME) algorithm [20], which has been widely used in 2D clock synthesis. The first phase in sDMBE is to determine the merging types and to construct the merging segments for each pair of subsets in a bottom-up traversal. Different from the existing 2D synthesis [23]–[25], which focused on slew-aware buffer insertion after clock routing, sDMBE performs buffer insertion during the bottom-up procedure. The goal of slew-aware buffering in sDMBE is to locate buffers while merging subsets, so that the load capacitances of buffers are within the given bound (CMAX). The impact of CMAX on the 3D clock slew is discussed in Section VI-E. Merging segments are obtained based on the merging distances, which are computed under the zero-skew equations in the Elmore delay model and wire length minimization goals. The second phase of sDMBE is to decide the exact locations of internal nodes, buffers, and TSVs in a top-down fashion and determine the routing topology of the overall clock nets. The complexity of our approach is $O(n)$, which makes it feasible for incremental clock routing or inclusion in a solution search framework.

Two samples of merging segments for unbuffered and buffered 3D clock trees are shown in Fig. 7. When merging child nodes u and v to parent node p, sDMBE first decides

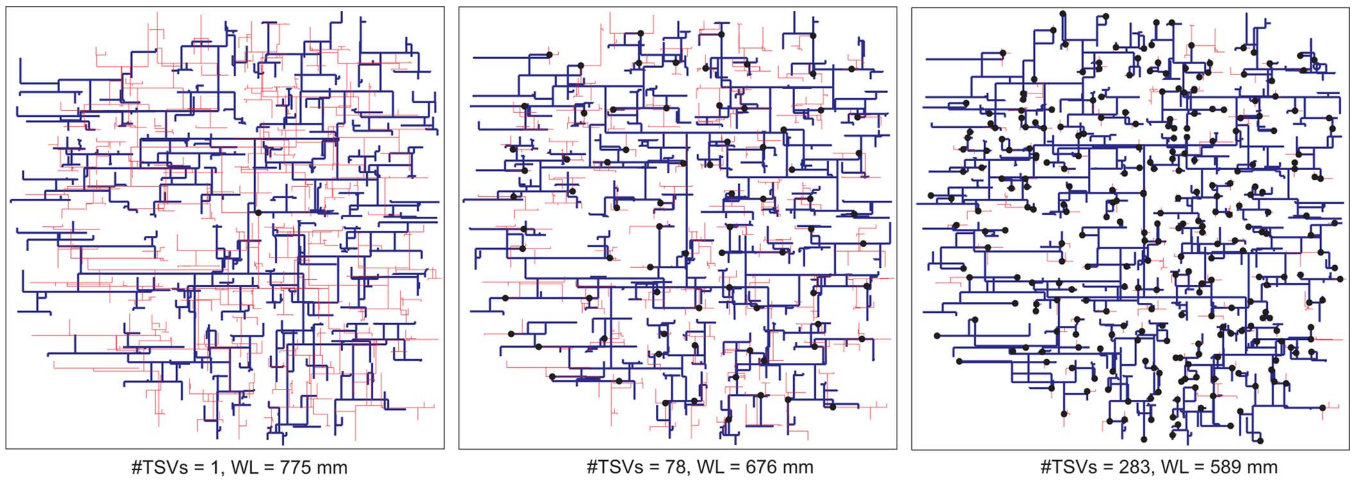

Fig. 8. 3D clock trees for the two-die stack r_3 with varying TSV bounds. The black dots are the TSV location candidates. And the bold and thin lines illustrate the clock nets in die-1 and die-2, respectively.

TABLE I
COMPARISON OF WIRE LENGTH (UM), POWER(MW), TSV COUNT (#TSVS), BUFFER COUNT (#BUFS), SIMULATION TIMES (#SIMS), TOTAL SIMULATION RUNTIME (S), AND SKEW (PS) BETWEEN THE EXHAUSTIVE SEARCH AND THE 3D-MMM-EXT ALGORITHM FOR TWO-DIE STACKS. THE TSV CAPACITANCE IS 15 FF, 50 FF, AND 100 FF

TSV Cap	ckt	Exhaustive search							3D-MMM-ext								
																Reduction (%)	
		#TSVs	WL	#Bufs	Power	Skew	#Sims	Runtime	#TSVs	WL	#Bufs	Power	Skew	#Sims	Runtime	WL	Power
15 fF	r_1	91	220362	275	0.122	14.6	37	602.5	93	221443	282	0.125	9.3	1	16.8	-0.5	-2.5
	r_2	222	433639	573	0.250	14.1	29	1059.5	211	445647	588	0.255	14.2	1	32.5	-2.8	-2.0
	r_3	320	582035	778	0.342	12.1	31	1712.3	297	583274	779	0.342	13.5	1	50.5	-0.2	0.0
	r_4	715	1157160	1587	0.696	16.1	41	4981.5	660	1165529	1594	0.698	16.8	1	107.1	-0.7	-0.3
	r_5	1129	1728660	2496	1.062	20.2	41	9104.3	1096	1737100	2509	1.065	19.8	1	187.7	-0.5	-0.3
50 fF	r_1	95	218257	292	0.129	11.9	37	623.5	85	221719	293	0.130	11.9	1	17.6	-1.6	-0.8
	r_2	222	438370	602	0.267	14.4	29	1087.8	205	448195	618	0.271	13.6	1	36.5	-2.2	-1.5
	r_3	253	605079	848	0.368	14.4	31	1508.0	288	589654	845	0.366	15.7	1	48.1	2.5	0.5
	r_4	660	1171810	1723	0.748	17.0	41	5391.1	639	1165253	1727	0.745	15.0	1	114.6	0.6	0.4
	r_5	1091	1753390	2726	1.155	18.3	41	9152.9	1020	1749543	2684	1.151	17.8	1	186.3	0.2	0.3
100 fF	r_1	56	230940	301	0.135	10.1	37	618.7	45	238242	303	0.137	12.6	1	16.0	-3.2	-1.5
	r_2	76	493957	654	0.284	13.3	29	1156.0	87	492966	661	0.287	13.0	1	33.5	0.2	-1.1
	r_3	60	674674	883	0.383	12.9	31	1733.6	112	645062	897	0.383	13.4	1	55.1	4.4	0.0
	r_4	254	1293830	1926	0.793	19.4	41	5798.7	247	1286784	1891	0.787	18.2	1	125.2	0.5	0.8
	r_5	250	2004250	2799	1.190	14.0	41	9323.5	328	1953453	2798	1.194	19.0	1	179.8	2.5	-0.3

the merging type based on the given 3D abstract tree and the CMAX constraint. Corresponding to the merging type among clock wires, buffers, and TSVs, we obtain the merging distances between nodes p and u, p and v in Fig. 7(a), the distances between node p and buffer b, buffer b and node u, nodes p, and v in Fig. 7(b).

V. EXTENSION OF 3D-MMM ALGORITHM

As illustrated earlier in Fig. 1, the overall wire length of the 3D clock tree reduces as more and more TSVs are used. Fig. 8 provides another demonstration that higher usage of TSVs leads to shorter wire length. This raises an important question: *what is the optimal number of TSVs for a 3D clock tree that leads to the minimum possible power consumption?* One obvious way to answer this question is by trying *all* possible TSV counts and choosing the best power result (an exhaustive search). This method, however, is very time consuming and requires prohibitive runtime as shown in Table I. Thus, our goal is to find this TSV count that leads to the minimum (or close-to-minimum) power result in much shorter runtime. This calls for careful attention to the impact of the TSV count not only on the overall wire length but also the total number of buffers and total TSV capacitance as these factors equally affect the overall power consumption.

We develop our new low-power 3D clock tree synthesis method, named 3D-MMM-ext, by extending our 3D-MMM algorithm presented in Section IV-B. The goal of the 3D-MMM-ext is to construct a low-power clock network by wisely assigning clock TSVs during the 3D abstract tree generation. In each top-down partition, let S be the current sink set. Let $Z(S)$ denote the vertical distance the set S spans, which can be expressed as

$$Z(S) = Z_{\max} - Z_{\min} \tag{2}$$

where $Z_{\max}$ and $Z_{\min}$ are the maximum and minimum die indices of the sinks within set S. Note that $Z(S)$ also indicates the minimum number of TSVs required by the clock network connecting all the sinks in S. Different from the 3D-MMM algorithm, which decides the cut direction (Z-cut or X/Y-cut) based on the TSV bound (lines 5 and 8 in Fig. 5), the key technique of 3D-MMM-ext is to determine the cutting orientation of

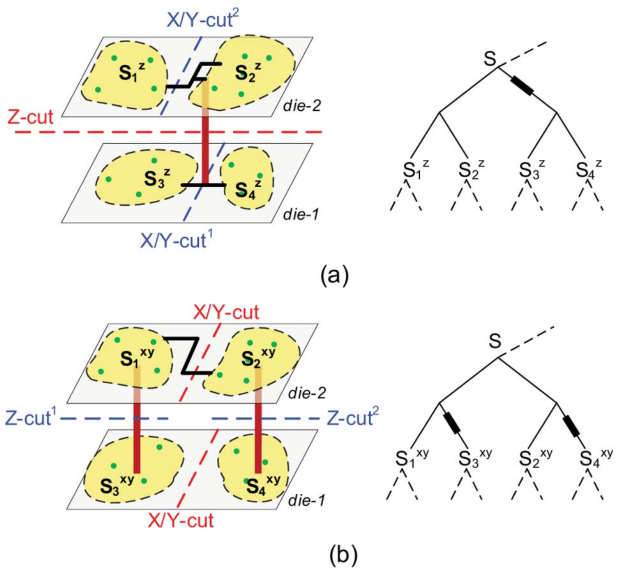

Fig. 9. The 3D-MMM-ext algorithm performed on a two-die stack with the sink set S. We show the 3D abstract trees, cut orders, and the subsets from case-1 and case-2 style partitions. (a) Case-1, where we apply Z-cut at the current iteration, and then X/Y-cut^1 and X/Y-cut^2 in die-1 and die-2, respectively. (b) Case-2, where we apply X/Y-cut at the current iteration, and then Z-cut^1 and Z-cut^2. P_z and P_{xy} are the cost of merging S_i^z and S_i^{xy} in (a) and in (b), respectively.

the current iteration (i.e., Z-cut or X/Y-cut) by looking ahead to the next cutting iteration, while estimating and comparing the costs of the following two cases.

- Case-1: apply Z-cut at the current iteration, and then apply X/Y-cut on each die once in the following iterations;
- Case-2: apply X/Y-cut at the current iteration, and postpone Z-cut to the next iteration.

Note that for the n-die stack case, Z-cut means applying die-wise partitions in multiple iterations until the sinks having the same die index are partitioned into the same subset. In the case-1 style partition, the sink set S has $stack(S) - 1$ times Z-cuts and $stack(S)$ times X/Y-cuts. S in the case-2 has one X/Y-cut and $2 \times (stack(S)-1)$ Z-cuts. Let S_i^z and S_i^{xy} represent the subsets after case-1 and case-2 style partitions, respectively. The sinks within the set S_i^z (or S_i^{xy}) are in the same die. Fig. 9 shows an example of determining the current cut direction using the 3D-MMM-ext on the sink set S. Fig. 9(a) shows the case-1 style partition, where Z-cut is applied during the current iteration and then X/Y-cut^1 and X/Y-cut^2 are applied on die-1 and die-2, respectively. Fig. 9(b) illustrates the case-2 partition results. We also show a part of the 3D abstract tree corresponding to case-1 and case-2 partitions, respectively. We have the following relation:

$$S = \bigcup_{i=1}^{4} S_i^z = \bigcup_{i=1}^{4} S_i^{xy}. \tag{3}$$

By comparing the cost of case-1 (P_z) and the cost of case-2 (P_{xy}), the cut direction of the current iteration is determined as follows:

$$\text{Current Cut} = \begin{cases} \frac{X}{Y}\text{-cut}, & \text{if } P_z > P_{xy} \\ Z\text{-cut}, & \text{otherwise.} \end{cases} \tag{4}$$

This means that if selecting Z-cut during the current iteration helps reduce power, then we choose Z-cut; otherwise, we choose X/Y-cut. The cost P_z is defined as follows:

$$P_z = \sum_{i \in \text{cond1}} P\left(S_i^z\right) + \sum_{j,k \in \text{cond2}} P\left(S_j^z, S_k^z\right). \tag{5}$$

Similarly

$$P_{xy} = \sum_{i \in \text{cond1}} P\left(S_i^{xy}\right) + \sum_{j,k \in \text{cond2}} P\left(S_j^{xy}, S_k^{xy}\right). \tag{6}$$

Let S_i represent either S_i^{xy} or S_i^z. The first item $P(S_i)$ in the cost function is the cost of the subset S_i, where cond1 covers the final subsets after the look-ahead partitions. The second item $P(S_j, S_k)$ in the cost function is the cost of connecting subsets S_j and S_k. $P(S_j, S_k)$ mainly comes from TSVs, global wires, and buffers. Therefore, cond2 covers all pairs of subtrees in the 3D abstract tree, where we merge those final subsets to their parent sink set S during the bottom-up traversal.

Considering the two-die stack examples in Fig. 9, P_z and P_{xy} can be expressed as follows:

$$\begin{aligned} P_z = & \sum_{i=1}^{4} P\left(S_i^z\right) + P\left(S_1^z, S_2^z\right) + P\left(S_3^z, S_4^z\right) \\ & + P\left(S_1^z \cup S_2^z, S_3^z \cup S_4^z\right) \end{aligned} \tag{7}$$

$$\begin{aligned} P_{xy} = & \sum_{i=1}^{4} P\left(S_i^{xy}\right) + P\left(S_1^{xy}, S_3^{xy}\right) + P\left(S_2^{xy}, S_4^{xy}\right) \\ & + P\left(S_1^{xy} \cup S_3^{xy}, S_2^{xy} \cup S_4^{xy}\right). \end{aligned} \tag{8}$$

To estimate the cost for each sink set, we use the half-parameter wire length model for $P\left(S_i^z\right)$ and $P\left(S_i^{xy}\right)$. Then, $P(S_j, S_k)$ is estimated as follows.

- If no TSV is required to connect S_j and S_k

$$P(S_j, S_k) \approx \text{CD}(S_j, S_k) \tag{9}$$

where $\text{CD}(S_j, S_k)$ is the distance between the centers of subsets S_j and S_k. In Fig. 9, $P\left(S_1^z, S_2^z\right)$, $P\left(S_3^z, S_4^z\right)$, and $P\left(S_1^{xy} \cup S_3^{xy}, S_2^{xy} \cup S_4^{xy}\right)$ belong to this case.

- If TSVs are needed to provide inter-die connection between S_j and S_k:

$$P(S_j, S_k) \approx \text{CD}(S_j, S_k) + \alpha \times \frac{C_{\text{TSV}}}{c} \tag{10}$$

where C_{TSV} is the TSV capacitance, c is the unit-length capacitance of the clock line, and α is an estimator representing the cost of TSV insertion. We use the following empirical equation to calculate α:

$$\alpha = (2 \times |Z(S_j) - Z(S_k)| + 3) \times \beta \tag{11}$$

where $\beta = 0.05$, 0.05 and 0.1 if the TSV capacitance is 15 fF, 50 fF, and 100 fF, respectively. In Fig. 9, $P\left(S_1^z \cup S_2^z, S_3^z \cup S_4^z\right)$, $P\left(S_1^{xy}, S_3^{xy}\right)$, and $P\left(S_2^{xy}, S_4^{xy}\right)$ belong to this case.

VI. Simulations and Discussions

We first examine a two-die stack to investigate the impact of the TSV count and TSV parasitics on clock power consumption.

Next, we show the efficiency of the 3D-MMM-ext algorithm in finding the optimal number of TSVs to be used for minimum power consumption. We then present the results of our clock slew control method. Lastly, we show the impact of scaling the supply voltage on 3D clock power consumption. We validate our claims with SPICE simulation results.

A. Simulation Settings

We construct a zero-Elmore-skew 3D clock network by using the 3D clock tree synthesis methods developed in Sections IV and V. We then extract the netlist of the entire 3D clock network for SPICE simulation. After the simulation, we obtain highly accurate power consumption and timing information of the entire clock network. Note that our 3D clock tree has zero skew under the Elmore delay model, but may have nonzero clock skew from SPICE simulation. Thus, we constrain the SPICE clock skew to be less than 3% of the clock period at a frequency of 1 GHz. The slew is constrained within 10% of the clock period. Clock power mainly comes from the switching capacitance of the interconnect, sink nodes, TSVs, and clock buffers.

The technical parameters are based on the 45 nm Predictive Technology Model [26]: per unit-length wire resistance is $0.1\Omega/\mathrm{um}$, and per unit-length wire capacitance is 0.2 fF/um. The buffer parameters are: driving resistance is 122 Ω, input capacitance is 24 fF, and intrinsic delay is 17 ps. The TSV resistance is 35 $\mathrm{m}\Omega$. In order to study the impact of the TSV *RC* parasitics on the 3D clock network, we vary the linear oxide thickness and choose three typical TSV capacitance values (i.e., 15 fF, 50 fF, 100 fF). The supply voltage is set to 1.2 V unless otherwise specified. The maximum load capacitance of each clock buffer, denoted CMAX, is set to 300 fF for slew control unless otherwise specified.

Our analysis focuses on two-die and six-die 3D clock networks. In the six-die case, the clock source is located in the middle die (die-3) as suggested in [10], unless otherwise specified. As a result, die-3 in a six-die clock network contains a complete tree. The IBM benchmarks r_1 to r_5 [27] are used. Since r_1 to r_5 are originally designed for 2D ICs, we randomly distribute the sinks into two or six dies. We then scale the footprint area by $\sqrt{N}$ to reflect the area reduction in the 3D design.

Sample clock trees in die-1 and die-3 of a six-die 3D clock network are shown in Fig. 10. The triangle denotes the clock source in die-3. Each die contains up to 20 TSVs. Note that die-3 has a single global tree that connects all the sinks, and die-1 contains multiple local trees that are connected to the clock source using TSVs.

B. Impact of TSV Count and Parasitic Capacitance

To investigate the impact of the TSVs on clock power consumption, we use a two-die stack implementation of the biggest benchmark r_5, which has 3101 sink nodes with input capacitances varying from 30 to 80 fF. Fig. 11 shows three clock power trend curves for a TSV capacitance (C_{TSV}) of 15 fF, 50 fF, 100 fF, respectively. On the x-axis we show the total number of TSVs used in each entire 3D clock tree, which is obtained by imposing a different TSV bound. Our baseline 3D clock network contains only one TSV between adjacent dies.

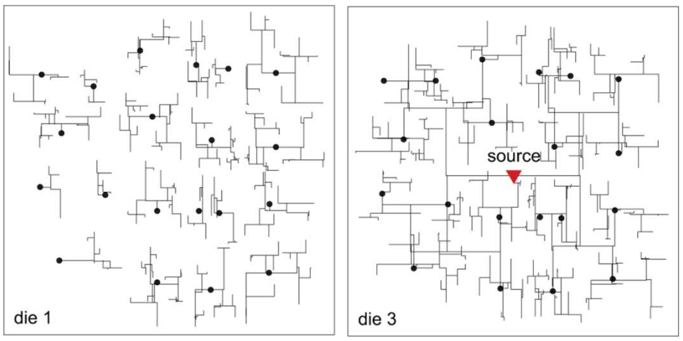

Fig. 10. Clock trees in die-1 and die-3 of a sample six-die 3D clock network, where the clock source is located in die-3. Black dots denote TSVs. The TSV bound is set to 20. Die-1 contains many local trees, whereas die-3 contains a single global tree.

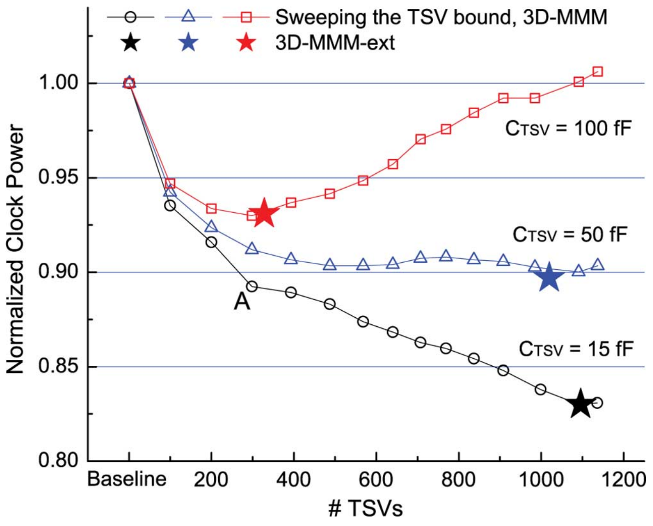

Fig. 11. Impact of the TSV capacitance and count on clock power for the two-die $r5$. The TSV capacitance (C_{TSV}) is set to 15 fF, 50 fF, and 100 fF. Our baseline is the clock tree that uses one TSV between adjacent dies. For each C_{TSV}, we show the 3D-MMM results by sweeping the TSV count. We also highlight the 3D-MMM-ext results for each C_{TSV}, which are marked as stars near to the trends.

The clock power is affected by both the TSV count and the TSV capacitance as shown in Fig. 11. First, using 15 fF TSVs in the clock network construction, the clock power decreases significantly when more TSVs are used. We are able to obtain a low-power clock network design by relaxing the TSV bound. We can achieve up to 17.0% power reduction as compared to the single-TSV case. The power savings mostly comes from wire length reduction, because the clock wire capacitance significantly affects the overall power consumed by the clock network. When more TSVs are used, the number of local trees in the non-source dies increases while their size decreases. This means that the multiple-TSV case encourages more local clock distribution in 3D designs while reducing the overall wire length. Second, if the TSV has a large capacitance (e.g., 50 fF, 100 fF), the contribution of the TSV capacitance to the overall power consumption is non-negligible. As a result, when the TSV count increases, the overall clock power reduction becomes slower.

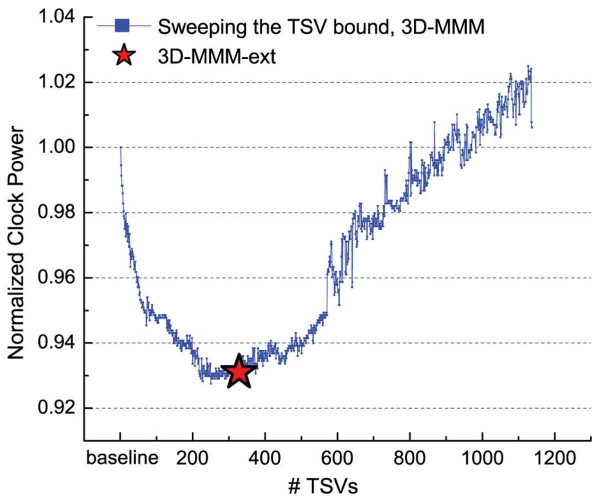

Fig. 12. Clock power trends for the two-die stack r_5 based on exhaustive search within the TSV count range [1, 1137]. The TSV capacitance is 100 fF. We also plot the 3D-MMM-ext algorithm result. The exhaustive search covers 1137 simulations on various clock trees. The runtime for each simulation is around 200 s.

Particularly, if the TSV capacitance is 100 fF, clock power does not decrease when the TSV count exceeds a certain amount and eventually starts increasing. In this case, the clock power from the TSV capacitance increases faster than the power decreases from wire length reduction.

From this trend study, we conclude that given a TSV parasitic capacitance, there exists an optimum number of TSVs that results in the minimum 3D clock power. This in turn allows us to choose the right TSV bound for a given power budget. If a power savings of 10% is required for using the 15 fF TSVs, the TSV bound of 300 can be used based on point A in Fig. 11.

C. Exhaustive Search Results

A straightforward way to find the "min-power TSV count," i.e., the number of TSVs used in a 3D clock tree that leads to the minimum overall clock power consumption, is to exhaustively sweep the TSV bound from 1 to infinity[3], constructing and simulating the entire 3D clock network corresponding to each TSV bound. By plotting the TSV count versus power trend curve, we are then able to find the optimum solution. Fig. 12 shows the clock power trend based on 1137 3D clock trees we generated and simulated for the two-die stack r_5. We assume the TSV parasitic capacitance is 100 fF.

We observe that the lowest power comes from the clock network that uses 250 TSVs, with 1.190 W clock power and 2 004 250 $\mu\mathrm{m}$ wire length. In addition, we observe that the exhaustive search result agrees with the TSV count versus power trend we presented in the previous section, although power fluctuates locally in a small range of the TSV count. If the TSV count exceeds 600, the clock power is much more sensitive to the TSV count increase. Using one more TSV may lead to the clock power increasing or decreasing by 1%. This is because, when using a large amount of TSVs, the clock network has a large number of smaller local trees. This means that the TSV capacitance itself is comparable to or even larger than that of a single local clock tree. In this case, using a few more TSVs leads to a large fluctuation in clock power.

[3]Note that the TSV bound of infinity means that we do not impose any restriction on the maximum number of TSVs used in each die. This usually results in a high usage of TSVs that mainly targets at wire length minimization.

The proposed exhaustive search method does allow us to find the min-power TSV count, but it is too costly in terms of runtime. The smaller step size we use for the TSV count in the search, the lower power of a 3D clock network we find, but more simulations as well as runtime are required. Note that the typical SPICE simulation time of a two-die r_5 clock network is around 200 s. Repeating this 1137 times is prohibitive.

D. 3D-MMM-ext Algorithm Results

In Fig. 12, the star indicates the solution obtained by our 3D-MMM-ext algorithm. Our algorithm does not involve any exhaustive search on the TSV count, but relies on our look-ahead based method to control the TSV usage and to minimize the overall power consumption. We observe that our 3D-MMM-ext generates a 3D clock tree that has a similar quality as the one obtained by the exhaustive research, but at a fraction of runtime. The runtime required for 3D-MMM-ext is comparable to that of generating a single 3D clock tree. The solution quality obtained by our 3D-MMM-ext algorithm can also be seen in Fig. 11, where the stars indicate the 3D trees produced by 3D-MMM-ext. The power consumption at these points is comparable to the minimum power solutions found in each curve.

Table I presents more detailed comparisons of wire length ($\mu\mathrm{m}$), buffer count (#Bufs), clock power (W), clock skew (ps), number of simulations (#sims), and the total simulation runtime (s) between the exhaustive search and the 3D-MMM-ext algorithm. We use two-die 3D stacks. We also show the wire length and power reduction of 3D-MMM-ext with respect to the exhaustive search. First, the clock power of 3D-MMM-ext is comparable to that of the exhaustive search. In most cases, 3D-MMM-ext has less than 1% power difference. In some cases, 3D-MMM-ext achieves even lower power (i.e., positive reduction) than the exhaustive search. This is mainly because the low-power design obtained by the exhaustive search depends on the sweeping granularity and simulation times. Second, the simulation runtime comparison reveals the effectiveness of our 3D-MMM-ext algorithm. 3D-MMM-ext requires only a single simulation, whereas the exhaustive search requires 29–41 simulations.

Tables II and III list the comparison between using a single TSV and using multiple TSVs (obtained with 3D-MMM-ext algorithm) cases. We use a two-die and a six-die implementation of our benchmark designs. First, the 3D-MMM-ext is able to find the low-power 3D clock trees. For the two-die stacks in Table II, the 3D-MMM-ext reduces the clock power by around 16.1%–18.8%, 10.3%–13.7%, and 6.6%–8.3% as compared with the single-TSV cases, and achieves wire length savings around 24.0%–26.5%, 23.9%–26.6%, and 16.6%–18.9%, when the TSV capacitance is 15 fF, 50 fF, and 100 fF, respectively. In the case of six-die stacks shown in Table III, our 3D-MMM-ext reduces power by up to 36.1%, 26.4%, and 9.1%, and reduces wire length by up to 50.7%, 47.4%, and 17.3%.

Table IV lists the comparisons between placing the clock source in die-1 and in die-3, for six-die stacks using the 3D-MMM-ext algorithm. When moving the clock source to the

TABLE II
COMPARISON OF WIRE LENGTH (UM), POWER (MW), TSV COUNT(#TSVS), BUFFER COUNT (#BUFS), SIMULATION RUNTIME (S), AND SKEW (PS) BETWEEN USING SINGLE TSV AND USING MULTIPLE TSVS (3D-MMM-EXT) FOR THE TWO-DIE STACKS. THE TSV CAPACITANCE IS 15 FF, 50 FF, AND 100 FF

TSV		Single TSV					Multiple TSVs (3D-MMM-ext)							
													Reduction (%)	
Cap	ckt	WL	#Bufs	Power	Skew	Runtime	#TSVs	WL	#Bufs	Power	Skew	Runtime	WL	Power
15 fF	r_1	291421	327	0.149	10.5	17.6	93	221443	282	0.125	9.3	16.8	24.0	16.1
	r_2	602484	706	0.314	15.4	43.2	211	445647	588	0.255	14.2	32.5	26.0	18.8
	r_3	775194	930	0.410	17.4	55.2	297	583274	779	0.342	13.5	50.5	24.8	16.6
	r_4	1586630	1990	0.855	18.2	122.8	660	1165529	1594	0.698	16.8	107.1	26.5	18.4
	r_5	2341420	2897	1.283	17.0	188.0	1096	1737100	2509	1.065	19.8	187.7	25.8	17.0
50 fF	r_1	291498	327	0.149	12.4	18.1	85	221719	293	0.130	11.9	17.6	23.9	12.8
	r_2	602485	706	0.314	15.2	38.4	205	448195	618	0.271	13.6	36.5	25.6	13.7
	r_3	775056	930	0.410	17.2	53.2	288	589654	845	0.366	15.7	48.1	23.9	10.7
	r_4	1586880	1991	0.855	14.8	121.5	639	1165253	1727	0.745	15.0	114.6	26.6	12.9
	r_5	2341360	2897	1.283	16.8	220.1	1020	1749543	2684	1.151	17.8	186.3	25.3	10.3
100 fF	r_1	291421	328	0.149	9.9	17.5	45	238242	303	0.137	12.6	16.0	18.2	8.1
	r_2	601929	707	0.313	13.5	40.0	87	492966	661	0.287	13.0	33.5	18.1	8.3
	r_3	775029	930	0.410	17.3	54.2	112	645062	897	0.383	13.4	55.1	16.8	6.6
	r_4	1586630	1992	0.855	15.7	131.3	247	1286784	1891	0.787	18.2	125.2	18.9	8.0
	r_5	2341460	2897	1.283	17.1	187.6	328	1953453	2798	1.194	19.0	179.8	16.6	6.9

TABLE III
COMPARISON OF WIRE LENGTH (UM), POWER (MW), TSV COUNT (#TSVS), BUFFER COUNT (#BUFS), SIMULATION RUNTIME (S), AND SKEW (PS) BETWEEN USING SINGLE TSV AND USING MULTIPLE TSVS (3D-MMM-EXT) FOR THE SIX-DIE STACKS. THE TSV CAPACITANCE IS 15 FF, 50 FF, AND 100 FF

TSV		Single TSV					Multiple TSVs (3D-MMM-ext, src in die-3)							
													Reduction (%)	
Cap	ckt	WL	#Bufs	Power	Skew	Runtime	#TSVs	WL	#Bufs	Power	Skew	Runtime	WL	Power
15 fF	r_1	272109	332	0.144	19.4	19.0	297	138223	214	0.092	12.8	10.5	49.2	36.1
	r_2	566944	684	0.298	16.1	45.0	668	280901	445	0.191	18.2	29.7	50.5	35.9
	r_3	717479	887	0.388	15.0	57.0	965	376634	626	0.264	17.1	45.8	47.5	32.0
	r_4	1496180	1870	0.816	18.5	119.8	2195	752370	1316	0.551	17.6	84.0	49.7	32.5
	r_5	2299220	2935	1.265	19.6	205.3	3497	1133262	2070	0.854	21.4	154.0	50.7	32.5
50 fF	r_1	272849	332	0.144	17.4	17.7	275	143626	257	0.106	18.5	11.5	47.4	26.4
	r_2	567686	684	0.299	15.0	46.6	631	302068	562	0.230	20.3	35.2	46.8	23.1
	r_3	719610	891	0.389	14.3	66.1	918	403235	775	0.316	18.5	50.2	44.0	18.8
	r_4	1493990	1870	0.815	15.0	123.0	2045	810708	1680	0.670	27.0	95.1	45.7	17.8
	r_5	2299590	2935	1.266	19.3	217.8	3270	1250269	2644	1.051	23.4	189.8	45.6	17.0
100 fF	r_1	273951	332	0.145	16.6	16.8	30	234821	309	0.133	29.0	17.1	14.3	8.3
	r_2	566803	685	0.298	11.1	45.1	80	468805	638	0.271	28.9	41.2	17.3	9.1
	r_3	720705	893	0.390	14.2	61.6	75	651298	873	0.374	23.1	60.3	9.6	4.1
	r_4	1497240	1873	0.817	14.0	126.5	115	1333034	1804	0.769	23.8	118.8	11.0	5.9
	r_5	2300620	2935	1.266	19.2	183.6	180	2014167	2780	1.179	28.3	186.7	12.5	6.9

TABLE IV
COMPARISON OF WIRE LENGTH (UM), POWER (MW), TSV COUNT (#TSVS), BUFFER COUNT (#BUFS), SIMULATION RUNTIME (S), AND SKEW (PS) BETWEEN CLOCK SOURCE LOCATING IN DIE-1 AND DIE-3 (BOTH USING 3D-MMM-EXT) FOR THE SIX-DIE STACKS. THE TSV CAPACITANCE IS 15 FF, 50 FF, AND 100 FF

TSV		Multiple TSVs (3D-MMM-ext, src in die-1)						Multiple TSVs (3D-MMM-ext, src in die-3)							
														Reduction (%)	
Cap	ckt	#TSVs	WL	#Bufs	Power	Skew	Runtime	#TSVs	WL	#Bufs	Power	Skew	Runtime	WL	Power
15 fF	r_1	375	141353	227	0.095	15.6	11.2	297	138223	214	0.092	12.8	10.5	2.2	3.2
	r_2	798	287536	479	0.197	20.0	32.3	668	280901	445	0.191	18.2	29.7	2.3	3.0
	r_3	1196	376081	665	0.268	14.5	49.6	965	376634	626	0.264	17.1	45.8	-0.1	1.5
	r_4	2594	766596	1371	0.561	17.5	95.4	2195	752370	1316	0.551	17.6	84.0	1.9	1.8
	r_5	4133	1167350	2174	0.876	20.2	163.4	3497	1133262	2070	0.854	21.4	154.0	2.9	2.5
50 fF	r_1	345	147503	284	0.113	16.0	3.8	275	143626	257	0.106	18.5	11.5	2.6	6.2
	r_2	742	309985	647	0.243	25.3	42.5	631	302068	562	0.230	20.3	35.2	2.6	5.3
	r_3	1063	423253	899	0.339	20.3	62.6	918	403235	775	0.316	18.5	50.2	4.7	6.8
	r_4	2335	856880	1931	0.719	24.4	113.8	2045	810708	1680	0.670	27.0	95.1	5.4	6.8
	r_5	3688	1349599	3086	1.143	25.5	195.8	3270	1250269	2644	1.051	23.4	189.8	7.4	8.0
100 fF	r_1	20	261396	322	0.143	15.4	17.1	30	234821	309	0.133	29.0	17.1	10.2	7.0
	r_2	40	537705	661	0.294	13.8	39.2	80	468805	638	0.271	28.9	41.2	12.8	7.8
	r_3	45	709790	902	0.393	14.1	54.2	75	651298	873	0.374	23.1	60.3	8.2	4.8
	r_4	90	1409870	1824	0.798	16.0	109.0	115	1333034	1804	0.769	23.8	118.8	5.4	3.6
	r_5	100	2154326	2827	1.225	17.1	180.2	180	2014167	2780	1.179	28.3	186.7	6.5	3.8

middle die (die-3), the 3D-MMM-ext achieves further power savings, especially in the case when the TSV capacitance is 100 fF. In addition, in most of the cases, e.g., the six-die stacks with 15 fF and 50 fF TSVs, the middle-die 3D-MMM-ext uses fewer TSVs and achieves lower power than the cases when the src is in die-1.

In most cases, the simulated clock skew is less than 20 ps, which is less than the 30 ps constraint. In the case of the six-die

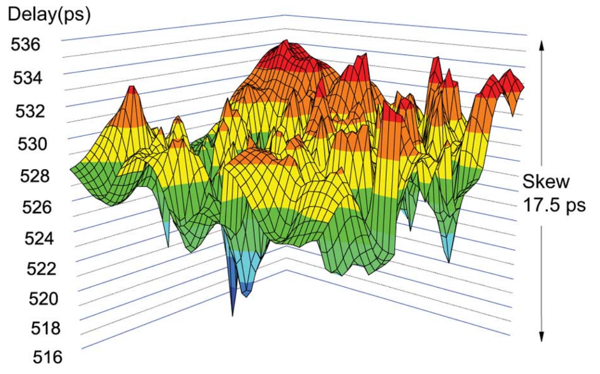

Fig. 13. Spatial distribution of propagation delay (ps) and clock skew (ps) of the clock source die, for the six-die stack r_5. The TSV count is 3497.

3D stack of r_5, Fig. 13 shows the spatial distribution of the propagation delay for the die containing the clock source. The TSV count is 3497. We observe that the clock skew among the six dies varies within $[17.5\ \mathrm{ps}, 21.4\ \mathrm{ps}]$. The skew of the entire 3D clock network is 21.4 ps. Referring to the TSV *RC* parasitics and the 300 fF CMAX constraint, the delay along each TSV is in the order of 0.01 ps. Compared with the > 500 ps src-to-sink delay, this means that the TSV itself contributes a negligible portion of delay to the entire src-to-sink delay. Note that our 3D clock tree synthesis algorithm builds a zero-skew tree under the Elmore delay model, which in practice shows discrepancy between SPICE simulation results.

E. Low-Slew 3D Clock Routing

Our goal in this experiment is to show that the TSV count also affects the clock slew distribution. Fig. 14 shows the slew distribution of the six-die 3D clock tree for r_5 among all sinks. The clock slew constraint is set to 100 ps, which is 10% of the clock period. The slew distribution of the single-TSV clock tree is shown in Fig. 14(a), whereas Fig. 14(b) shows the slew distribution of the multiple TSV clock tree using the 3D-MMM-ext. In the single-TSV clock tree, slew varies within $[34.2\ \mathrm{ps}, 82.7\ \mathrm{ps}]$ with an average slew of 53.9 ps. The slew distribution of the multiple-TSV case is in the range of $[29.1\ \mathrm{ps}, 80.3\ \mathrm{ps}]$ with an average slew of 46.8 ps. Compared with the single-TSV case, the multiple-TSV case reduces the maximum slew and average slew by 2.4 and 7.1 ps, respectively. The main reason for the improved slew distribution of the multiple-TSV 3D tree is the shorter wire length, which in turn reduces the capacitive load. Thus, we conclude that multiple TSVs are effective in improving the slew distribution.

The impact of CMAX, the maximum clock buffer load capacitance, on slew variations (min, average, max) and power consumption in the single-TSV and multiple-TSV clock trees is shown in Fig. 15. First, CMAX remains an efficient means to control the maximum slew in 3D clock network design. Both the single-TSV and multiple-TSV cases have similar trends as CMAX varies from 300 to 175 fF: a smaller CMAX reduces the maximum slew, but increases the clock power. This is because each buffer stage is allowed to drive a smaller capacitance with smaller CMAX, which in turn requires more buffers and thus consumes more power. Second, given a certain CMAX, multiple-TSV clock trees always have reduced maximum slew

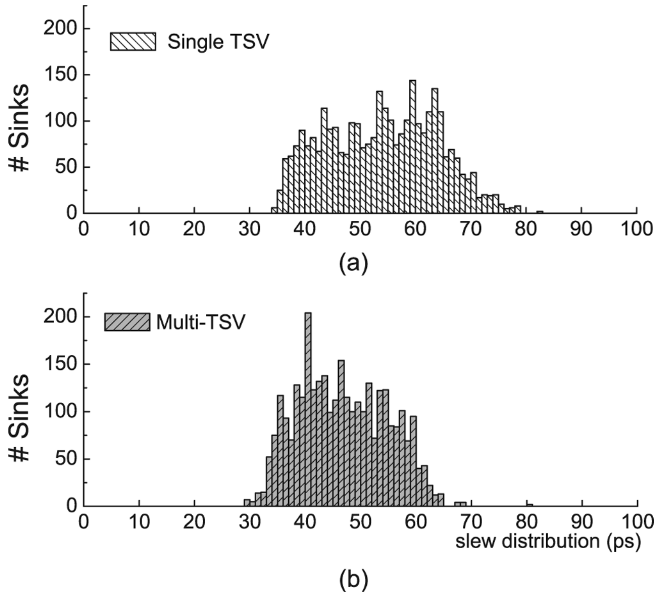

Fig. 14. Slew distribution of six-die 3D clock network among all sinks. Slew constraint is set to 10% of the clock period, and CMAX is 300 fF. (a) Slew distribution in the single-TSV clock tree, (b) in the multiple-TSV clock tree.

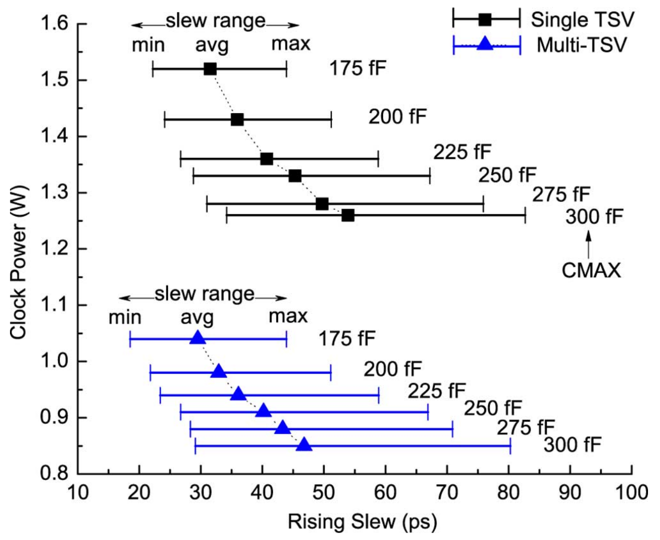

Fig. 15. Slew variations and power comparisons between single-TSV and multiple-TSV clock trees. CMAX varies from 175 to 300 fF.

and less average slew, as compared with the single-TSV cases. Third, we note that the multiple-TSV case always consumes less power than the single-TSV case. Therefore, we conclude that the multiple-TSV case achieves both low power and better slew results.

F. Scaling the Supply Voltage

In this section, we investigate the impact of supply voltage scaling on 3D clock power, clock skew, and slew. The clock skew and power changes when the supply voltage is scaled down from 1.2 to 0.7 V. These changes are shown in Fig. 16, for a clock frequency of 1 GHz. We first compare the two clock networks based on 15 fF and 100 fF TSV capacitance. Both of the clock networks use 125 TSVs. We first observe that both clock networks have a similar trend when the supply voltage is scaled

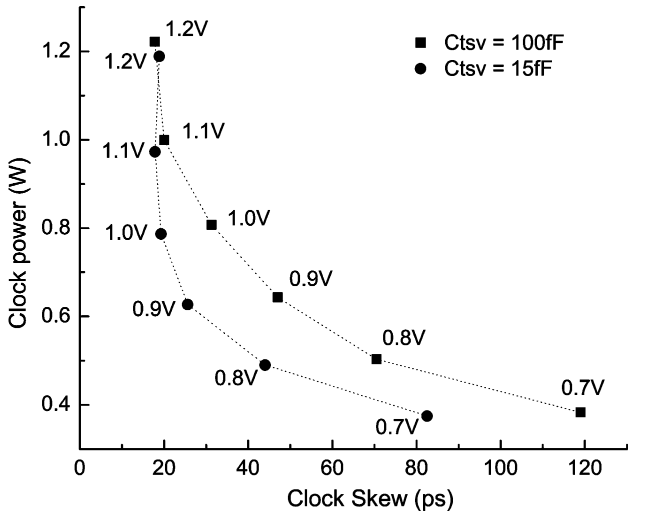

Fig. 16. Impact of scaling the supply voltage on clock power and clock skew. The supply voltage decreases from 1.2 to 0.7 V. We compare two clock networks using 15 and 100 fF TSVs. Each network uses 125 TSVs.

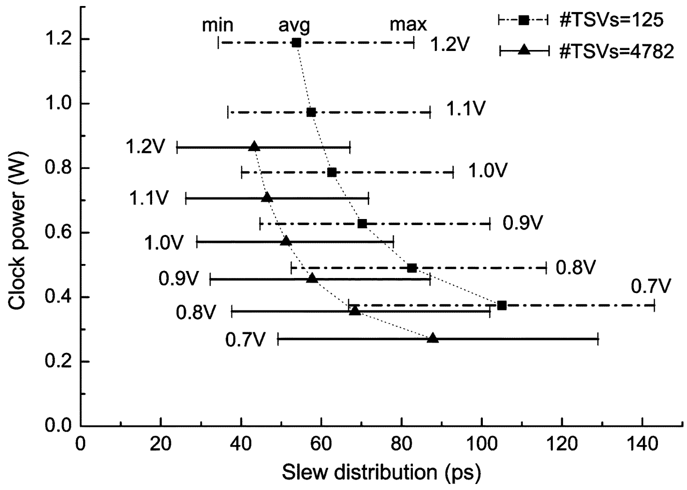

Fig. 17. Impact of scaling the supply voltage on the clock slew distribution and clock power. Supply voltage decreases from 1.2 to 0.7 V. The TSV capacitance is 15 fF. We compare two clock networks using 125 TSVs and 4782 TSVs.

down: The clock power is reduced from around 1.2 W to 0.4 W, which is more than a 65% power reduction. Second, the clock skew increases from 20 to 80 ps if the TSV capacitance is 15 fF, and from 20 to 120 ps for 100 fF TSVs. Moreover, the clock skew for a 100 fF TSV capacitance increases faster than that for a 15 fF TSV capacitance. This is mainly because the former uses 2830 clock buffers, whereas the latter uses 2789 clock buffers. The more buffers a 3D clock tree contains, the faster the clock skew degrades with the supply voltage scaling down. Thus, if the maximum simulated clock skew is set to 40 ps, the clock network can normally operate above 0.8 V and 0.9 V, using 15 fF TSVs and 100 fF TSVs, respectively.

The impact of scaling the supply voltage on the clock slew distribution and power changes is shown in Fig. 17. The supply voltage is scaled down from 1.2 to 0.7 V, and the clock frequency is kept at 1 GHz. We compare two clock networks: the first uses 125 TSVs, and the second uses 4782 TSVs. Both clock networks are based on 15 fF TSVs. We find that the clock network using 4782 TSVs always has better control on slew distribution, regardless of the supply voltage value. In addition, the clock tree using 4782 TSVs consumes lower power than the tree using 125 TSVs for all the voltage levels. As discussed earlier, this is due to the faster reduction in capacitance from shorter wire length and fewer buffers than the TSV capacitance increase by using more TSVs. When the supply voltage scales down, the power difference between these two clock networks is reduced.

TABLE V
COMPARISONS WITH [11]

	MMM-3D+ZCTE-3D [11]			Ours		
ckt	#TSVs	WL	Delay	#TSVs	WL	Delay
r_1	83	1441849	1.64	55	1521459	1.68
r_2	197	2831346	4.34	155	2978537	4.33
r_3	276	3725294	6.37	214	3918503	6.51
r_4	653	7424886	19.28	510	7856725	19.43
r_5	1052	10940984	35.20	811	11528598	35.94

G. Comparison With Existing Work

We show the comparison of our work with [11] in Table V. Note that [11] does not support buffer insertion or provide any SPICE simulation results. However, we attempted a comparison with [11] by disabling our buffer insertion. We use the same benchmark settings and report the skew and delay values in the Elmore delay model. We observe that our method uses 21.3%–33.7% fewer TSVs than [11] while using 5.2%–5.8% more wire length. Note that in our work we can control the TSV count versus wire length tradeoff by tweaking the TSV bound. In addition, these results come from unbuffered clock trees. Our sDMBE algorithm supports buffer insertion, which helps to properly control wire snaking and therefore better minimizes the wire length.

VII. CONCLUSION

In this paper, we explored design optimization techniques for reliable low-power and low-slew 3D clock network design. We thoroughly studied the impact of the TSV count and the TSV capacitance on clock power trends. We observed that using more TSVs helps reduce the wire length and power consumption; and shows better control over clock slew variations. However, in the case of a large TSV parasitic capacitance, clock power could increase if too many TSVs are used. We also observed that a smaller maximum loading capacitance on the clock buffers efficiently lowers the 3D clock slew. Furthermore, we developed a low-power 3D clock tree synthesis algorithm called 3D-MMM-ext. Experimental results show that our 3D-MMM-ext algorithm constructs low-power 3D clock designs that have comparable power and reliability to an exhaustive search with a few orders of magnitude shorter runtime.

REFERENCES

[1] International Technology Roadmap for Semiconductors (ITRS) [Online]. Available: http://www.itrs.net

[2] P. J. Restle, T. G. McNamara, D. A. Webber, P. J. Camporese, K. F. Eng, K. A. Jenkins, D. H. Allen, M. J. Rohn, M. P. Quaranta, D. W. Boerstler, C. J. Alpert, C. A. Carter, R. N. Bailey, J. G. Petrovick, B. L. Krauter, and B. D. McCredie, "A clock distribution network for microprocessors," *IEEE J. Solid-State Circuits*, vol. 36, no. 5, pp. 792–799, May 2001.

[3] E. G. Friedman, "Clock distribution networks in synchronous digital integrated circuits," *Proc. IEEE*, vol. 89, pp. 665–692, May 2001.

[4] Q. K. Zhu, *High-Speed Clock Network Design*. New York: Springer, 2003.

[5] J. U. Knickerbocker, P. S. Andry, B. Dang, R. R. Horton, M. J. Interrante, C. S. Patel, R. J. Polastre, K. Sakuma, R. Sirdeshmukh, E. J. Sprogis, S. M. Sri-Jayantha, A. M. Stephens, A. W. Topol, C. K. Tsang, B. C. Webb, and S. L. Wright, "Three-dimensional silicon integration," *IBM J. Res. Develop.*, vol. 52, no. 6, pp. 553–569, 2008.

[6] J. Vardaman, 3-D through-silicon vias become a reality 2007 [Online]. Available: http://www.semiconductor.net/article/CA6445435.html

[7] S. L. Wright, P. S. Andry, E. Sprogis, B. Dang, and R. J. Polastre, "Reliability testing of through-silicon vias for high-current 3D applications," in *Proc. Electron. Compon. Technol. Conf. (ECTC)*, 2008, pp. 879–883.

[8] J. Minz, X. Zhao, and S. K. Lim, "Buffered clock tree synthesis for 3D ICs under thermal variations," in *Proc. Asia South Pacific Design Automat. Conf.*, 2008, pp. 504–509.

[9] X. Zhao, D. L. Lewis, H. H. S. Lee, and S. K. Lim, "Pre-bond testable low-power clock tree design for 3D stacked ICs," in *Proc. IEEE Int. Conf. Computer-Aided Design*, 2009, pp. 184–190.

[10] X. Zhao and S. K. Lim, "Power and slew-aware clock network design for through-silicon-via (TSV) based 3D ICs," in *Proc. Asia South Pacific Design Automat. Conf.*, 2010, pp. 175–180.

[11] T.-Y. Kim and T. Kim, "Clock tree embedding for 3D ICs," in *Proc. Asia South Pacific Design Automat. Conf.*, 2010, pp. 486–491.

[12] V. F. Pavlidis, I. Savidis, and E. G. Friedman, "Clock distribution networks for 3-D integrated circuits," in *IEEE Custom Integrated Circuits Conf. (CICC)*, 2008, pp. 651–654.

[13] V. Arunachalam and W. Burleson, "Low-power clock distribution in a multilayer core 3D microprocessor," in *Proc. 18th ACM Great Lakes Symp. VLSI*, 2008, pp. 429–434.

[14] D. H. Kim, K. Athikulwongse, and S. K. Lim, "A study of through-silicon-via impact on the 3D stacked IC layout," in *Proc. IEEE Int. Conf. Computer-Aided Design*, 2009, pp. 674–680.

[15] R. Weerasekera, M. Grange, D. Pamunuwa, H. Tenhunen, and L.-R. Zheng, "Compact modeling of through-silicon vias (TSVs) in three-dimensional (3-D) integrated circuits," in *IEEE Int. Conf. 3D System Integration (3DIC)*, 2009, pp. 1–8.

[16] I. Savidis and E. G. Friedman, "Closed-form expressions of 3-D via resistance, inductance, and capacitance," *IEEE Trans. Electron Devices*, vol. 56, no. 9, pp. 1873–1881, Sep. 2009.

[17] G. Katti, M. Stucchi, K. De Meyer, and W. Dehaene, "Electrical modeling and characterization of through silicon via for three-dimensional ICs," *IEEE Trans. Electron Devices*, vol. 57, no. 1, pp. 256–262, Jan. 2010.

[18] T. Bandyopadhyay, R. Chatterjee, D. Chung, M. Swaminathan, and R. Tummala, "Electrical modeling of through silicon and package vias," in *IEEE Int. Conf. 3D System Integrat. (3DIC)*, Sep. 2009, pp. 1–8.

[19] W. C. Elmore, "The transient analysis of damped linear networks with particular regard to wideband amplifiers," *J. Appl. Phys.*, vol. 19, no. 1, pp. 55–63, 1948.

[20] K. D. Boese and A. B. Kahng, "Zero-skew clock routing trees with minimum wirelength," in *Proc 5th Annu. IEEE Int. ASIC Conf. Exhibit*, 1992, pp. 17–21.

[21] M. Jackson, A. Srinivasan, and E. Kuh, "Clock routing for high-performance ICs," in *Proc. ACM Design Automat. Conf.*, 1990, pp. 573–579.

[22] J.-S. Yang, K. Athikulwongse, Y.-J. Lee, S. K. Lim, and D. Z. Pan, "TSV stress aware timing analysis with applications to 3D-IC layout optimization," in *Proc. ACM Design Automat. Conf.*, 2010, pp. 803–806.

[23] G. E. Tellez and M. Sarrafzadeh, "Minimal buffer insertion in clock trees with skew and slew rate constraints," *IEEE Trans. Computer-Aided Design Integrated Circuits Syst.*, vol. 16, no. 4, pp. 333–342, Apr. 1997.

[24] C. Albrecht, A. B. Kahng, B. Liu, I. I. Mandoiu, and A. Z. Zelikovsky, "On the skew-bounded minimum-buffer routing tree problem," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 22, no. 7, pp. 937–945, Jul. 2003.

[25] S. Hu, C. J. Alpert, J. Hu, S. K. Karandikar, Z. Li, W. Shi, and C. N. Sze, "Fast algorithms for slew-constrained minimum cost buffering," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 26, no. 11, pp. 2009–2022, Nov. 2007.

[26] Predictive technology model [Online]. Available: http://ptm.asu.edu/

[27] GSRC Benchmark [Online]. Available: http://vlsicad.ucsd.edu/GSRC/bookshelf/Slots/BST

Xin Zhao (S'07) received the B.S. degree from the Electronic Engineering Department, Tsinghua University, in 2003, and the M.S. degree from the Computer Science and Technology Department, Tsinghua University, in 2006. She is currently working toward the Ph.D. degree in the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta.

Her research interests include computer-aided design for VLSI circuits, especially on physical design for low power, robustness, and 3D ICs.

Ms. Zhao was the recipient of the Best Paper Award Nomination at the International Conference on Computer-Aided Design in 2009.

Jacob Rajkumar Minz (S'05–M'06) received the B.Tech. degree in Computer Science and Engineering from the Indian Institute of Technology (IIT), Kharagpur, India, in 2001, and the Ph.D. degree in electrical and computer engineering from Georgia Institute of Technology, Atlanta, in 2006.

He was with the Advanced VLSI Design Laboratory, IIT Kharagpur, in 2001 for a year, where he was involved in the design of digital chips. He is now employed in Synopsys Inc., as a Senior Research Development Engineer since 2006, and currently located in Bangalore, India. His areas of interest are physical-aware logic synthesis, and optimization algorithms for electronic design automation.

Sung Kyu Lim (S'94–M'00–SM'05) received the B.S., M.S., and Ph.D. degrees from the Computer Science Department, University of California, Los Angeles (UCLA), in 1994, 1997, and 2000, respectively.

In 2001, he joined the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, where he is currently an Associate Professor. His research focus is on the architecture, circuit, and physical design for 3D ICs and 3D System-in-Packages. He is the author of *Practical Problems in VLSI Physical Design Automation* (Springer, 2008).

Dr. Lim received the Design Automation Conference (DAC) Graduate Scholarship in 2003 and the National Science Foundation Faculty Early Career Development (CAREER) Award in 2006. He was on the Advisory Board of the ACM Special Interest Group on Design Automation (SIGDA) during 2003–2008 and received the ACM SIGDA Distinguished Service Award in 2008. He was an Associate Editor of the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMS during 2007–2009 and served as a Guest Editor for the ACM Transactions on Design Automation of Electronic Systems (TODAES). He has served the Technical Program Committee of several ACM and IEEE conferences on electronic design automation. He is a member of the Design International Technology Working Group for the 2009 renewal of the International Technology Roadmap for Semiconductors (ITRS).