Clock Clustering and IO Optimization for 3D Integration

Samyoung Bang*, Kwangsoo Han†, Andrew B. Kahng†‡ and Vaishnav Srinivas†

†ECE and ‡CSE Departments, UC San Diego, La Jolla, CA 92093

*Samsung Electronics Co. Ltd, Hwaseong-si, South Korea

eva.bang@samsung.com, {kwhan, abk, vaishnav}@ucsd.edu

***Abstract*—3D interconnect between two dies can span a wide range of bandwidths and region areas, depending on the application, partitioning of the dies, die size, and floorplan. We explore the concept of dividing such an interconnect into local *clusters*, each with a *cluster clock*. We combine such clustering with a choice of three clock synchronization schemes (synchronous, source-synchronous, asynchronous) and study impacts on power, area and timing of the clock tree, data path and 3DIO. We build a model for the power, area and timing as a function of key system requirements and constraints: total bandwidth, region area, number of clusters, clock synchronization scheme, and 3DIO frequency. Such a model enables architects to perform pathfinding exploration of clocking and IO power, area and bandwidth optimization for 3D integration.**

***Keywords:* 3D IO, 3D integration, clock distribution, synchronization**

I. Introduction

3D interconnect can span a wide range of bandwidths (few GB/s to hundreds of GB/s) and region area (few mm^2 to the full die area). The 3DIO *frequency* and *synchronization scheme* are two key choices that play into the power and area for such an interconnect. Clock and data power for such interconnect can often be a significant component of total chip power. The synchronization scheme for the 3DIO clocking could be (i) *synchronous* (traditional clock tree), (ii) *source-synchronous* (forwarded clock), or (iii) *asynchronous* (separate clock trees). Based on the 3DIO frequency and synchronization scheme, the design could lend itself to dividing the clock into local *clusters* as shown in Figure 1, which are tightly skew-balanced within themselves but have looser skew requirements between clusters. This allows for a tighter timing budget for the clock and data paths and also lends itself to use of low-power 3DIO for source-synchronous and asynchronous schemes at the expense of overhead in the clock generation and distribution for every cluster. The 3DIO frequency that can be achieved depends on the timing budget, which in turn depends on the synchronization scheme and clustering.

In this work, we investigate the optimal choice of 3DIO frequency, synchronization scheme and clustering for 3D interconnect power, area and bandwidth optimization. The optimal 3DIO frequency depends on the cost function of power and area. While wider and slower buses generally consume less power due to lower clock tree power and lower 3DIO power, they do take up more area, especially due to limitations in the micro-bump pitch and 3DIO electrostatic discharge (ESD) requirements.

The default clocking methodology for 3D is likely still based on to be a synchronous clocking scheme. Previous research has sought to improve CTS outcomes and 3D clock tree structures for such a clocking scheme [18], [17], [2], [7], [10]. These studies on 3D clock trees focus on optimization of the synchronous clock tree to minimize skew, power and area, but do not consider alternative synchronization schemes. In this paper, we study different synchronization and clustering schemes, seeking to identify and model the optimal choice of 3DIO frequency (number of 3DIOs), clock synchronization scheme, and clustering for a given cost function of power, area and bandwidth.

To find these optimal choices, we build several analytical models to estimate the power, area and timing of the clock tree, data path and 3DIO. The models that we develop encompass on-die clock tree and data path models, and the 3DIO models from [4], adapted for the three synchronization schemes. We combine analytical models [12], [13], [23] and metamodeling techniques [3] to fit models against data from a Design of Experiments (DOE).

Our contributions in this paper are as follows.

- We propose a new view of the design space for 3DIO frequency, clock synchronization scheme and clustering based on bandwidth and region area of the interconnect.
- We develop accurate power and area model for the 3D interconnect based on bandwidth, region area, number of clusters, cluster clock frequency, and clock synchronization scheme.
- We demonstrate how the space of design requirements and constraints is partitioned according to the choice of optimal synchronization scheme.

In the following, Section II describes the concept of clustered clocking, elaborates on the differences between the clocking schemes, and provides hypotheses on the design space partitioning based on these options. In Section III, we propose a methodology to build power and area models for the design space based on DOE results, using metamodeling techniques [3]. This modeling can then enable a flow that chooses the optimal choice of 3DIO frequency, synchronization scheme and clustering as described in Section II. The P&R and timing flow used to implement our design of experiments (DOE) is described in Section IV. Section V describes our DOE and the results, including the fitted models versus the DOE data. We summarize the paper in Section VI.

II. 3DIO Clustering and Clocking

One way to localize the clock tree of the 3D interconnect is to divide it into clusters. Such a clustered interconnect has a *cluster clock*, with skew constraints that could be tighter than those of the *global clock*. Figure 1 shows the design divided into four clusters. The global clock is distributed to the clusters, and once in the cluster, a *cluster clock tree* distributes the clock within the cluster. The skew constraints for the 3D interconnect are limited to the cluster clock tree; hence, use of the cluster clock tree allows for smaller skew. Further, the 3DIO array is now divided per cluster as well, which means that the data paths, going from the uniformly distributed sinks to the 3DIO at the center of the cluster, are much shorter. On the other hand, clustering the design has multiple overheads: synchronizing data between clusters; the clock 3DIO per cluster to the top die; and the blockages caused by the 3DIO array being distributed per cluster.

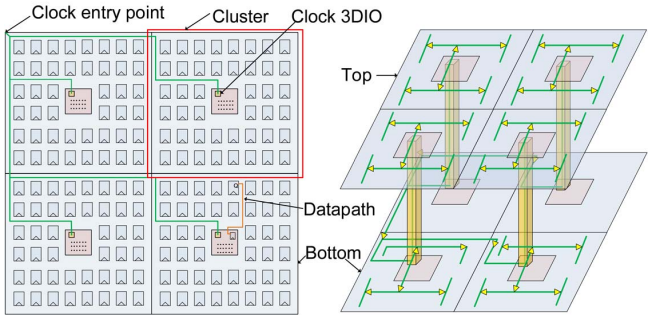

Fig. 1. Clustering the clock domains.

The cluster clock between the two dies must be synchronized. There are three basic methods used to synchronize the interconnect.

(i) Synchronous. In the synchronous method, the cluster parts in both dies share a common cluster clock tree that is balanced to end points in both dies. The clock tree on each die will vary according to the process variation of that die, so inter-die skew is large for the inter-die tree. But, at lower frequencies, this scheme provides a simple approach that is consistent with low-power implementation.
(ii) Source-synchronous. In the source-synchronous method, a clock is forwarded from the bottom die to the top die (or vice-versa). This forwarded clock is then built into a tree for the sinks on the top die. Such a scheme does not require skew balancing across two dies, but requires *balance delays (T_b)* within each die on the data path to match the clock insertion delay and enable source-synchronous data capture. There are power and area overheads associated with such balancing, but the improvement in the timing budget means that higher speeds can be achieved over the 3DIO. Both synchronous and source-synchronous data paths can enable a double data rate (DDR, both edges of the clock) design without any overhead in clock generation. We assume such a DDR design since it provides a way to minimize the number of 3DIOs.
(iii) Asynchronous. The asynchronous scheme has a FIFO that helps clock domain crossing between the two dies. Since the latency impact can be large, this is often combined with a high-speed serial IO that enables lower latency. As a consequence of the serial IO, we also obtain a much smaller IO footprint (smaller number of 3DIOs) at the expense of power and latency. We assume a 1:8 serialization for the asynchronous scheme. Although further serialization may be possible, the latency impact for much larger serialization ratios could be considerable, given the need to deserialize as well as the much slower cluster clock needed for a given 3DIO maximum frequency.

We observe that the source-synchronous clocking scheme allows for higher 3DIO frequencies than the synchronous clocking scheme, as it matches the delays in each die and does not expose the 3DIO timing to inter-die variations in the clock tree. But, such matching delays come with an area and power cost. Asynchronous clocking schemes provide a unique way of improving power for higher 3DIO frequencies. As they serialize the bus before off-chip transmission, such clocking schemes allow the cluster clock to operate at a lower speed while achieving a higher 3DIO bandwidth. This enables cluster clock tree power reduction at higher 3DIO speeds, offsetting the increased 3DIO power incurred by a narrower and faster bus. Furthermore, the narrower bus itself allows for 3DIO area reduction. When asynchronous clocking is combined with clustering, the 3DIO timing budget enables higher 3DIO frequencies, allowing for interesting tradeoffs between power and area. Exposing how these choices affect power and area based on the bandwidth and region area can enable system architects to make better 3DIO and clocking choices.

The three synchronization schemes are outlined in Figure 2. The figure shows the DDR (double data rate) implementation for the synchronous and source-synchronous schemes, which effectively doubles the 3DIO frequency and implies a half-cycle timing path for setup (as opposed to a full-cycle timing path as in a traditional single data rate design). The data path is shown in red, while the clock path in blue. For the asynchronous case, the clock and data path is common, shown in purple. The bottom die clock tree is shown in yellow, while the top die clock tree is shown in light blue. As can be seen, the synchronous case exposes an inter-die skew between the yellow and light-blue clock trees, while the source-synchronous case does not. Further description of Figure 2 can be found in Section III.

For a given 3DIO topology (defined by the synchronization scheme and number of clusters) there is a maximum frequency that can be achieved by the 3DIO. This maximum frequency is determined by two factors: (i) the 3DIO timing budget for retiming across the dies, and (ii) the on-die timing budget for retiming from the launch sinks to the 3DIO (or vice-versa from the 3DIO to the capture sinks).

In general, we expect the maximum 3DIO frequency to be the highest for asynchronous/highly-clustered and lowest for synchronous/lightly-clustered topologies. This is because on-die and

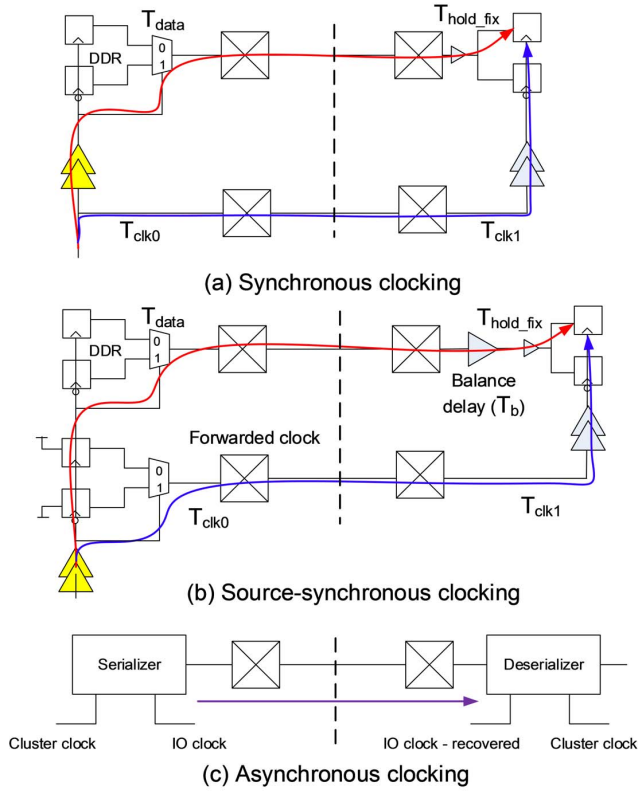

Fig. 2. Clock synchronization schemes.

3DIO timing budgets worsen as we progress from the former end of the design space to the latter end. The 3DIO timing budget is best for asynchronous schemes due to embedded clocks not having tight skew requirements, and much better jitter tolerance. The on-die timing budget is also better for asynchronous schemes due to lower cluster clock frequency (a consequence of serial 3DIO), and properties of the local sinks (together, highly-clustered).

In the following sections, we build our model and review the results in the context of three problem statements:

1) For a given BW and region area, find a clock synchronization scheme, number of clusters and clock frequency that minimizes power, given an area constraint.
2) For a given BW and region area, find a clock synchronization scheme, number of clusters and clock frequency that minimizes area, given a power constraint.
3) For a given region area, find a clock synchronization scheme, number of clusters and clock frequency that maximizes BW for a given power and area constraint.

III. Power, Area and Timing Model

In this section, we describe our model which accurately estimates total power, area and timing (worst negative slack (WNS)) for a given tree topology (synchronization scheme and clusters), bandwidth and region area. Figure 3 shows the *3DIO/CTS directed graph* of our modeling approach. The primary inputs (indicated by circles) are bandwidth, region area, cluster clock frequency, synchronization scheme and number of clusters. Based on these inputs, we calculate the number of sinks/FFs ($=2 \cdot BW/f$), 3DIO frequency ($f_{IO} = 2 \cdot f$ for synchronous and source-synchronous schemes; $f_{IO} = 8 \cdot f$ for asynchronous scheme), number of 3DIOs ($=BW/f_{IO}$), and maximum skew and transition requirements ($=k/f$).

The analytical expressions given below are often hard to fit across large ranges of many input parameters. We employ metamodeling techniques to improve fitting across the large ranges and number of

input variables. We choose to use an ANN (Artificial Neural Network) model for our fit [3]. Feeding just the primary inputs in Figure 3 does not, in our experience, yield very good results; however, with the help of the directed graph, a systematic approach to propagate the inputs through the graph aids the ANN model in finding fits for power, area and WNS within +/-15% across our design of experiments (DOE). In the following subsections, we describe the details of our model. Section V then gives fitting results. Note that to estimate the final outcomes (i.e., power, area, and WNS), we need buffer and wirelength models for clock and data path as our foundations.

A. Clock and Data Wirelength

The work of Snyder et al. [21] and Steele and Snyder [22] show that the expected total tree length of a Euclidean Steiner minimal tree over n points uniformly distributed within a bounded region A_{reg} is proportional to $\sqrt{A_{reg} \cdot N}$ for sufficiently large N. The constant of proportionality, denoted by k, is dependent on the functional of the pointset. Further, we empirically observe that the constant is not a single value, but is itself a function of region area A_{reg}, number of clusters N_c, and maximum transition time constraint T_{tran}^{max} ($k = f(A_{reg}, T_{tran}^{max}, N_c)$).

Clock wirelength significantly changes according to the clock tree structure (tall and narrow tree vs. short and wide tree). To analyze the structure of the clock tree generated by commercial tools, we extract the average fanout and driven wirelength of clock buffers at each depth (l) of clock trees synthesized with different region areas and/or number of FFs (N_{ff}). The depth of clock source is zero and the maximum depth is L, which is the number of buffer stages on the longest path from the clock source to any sink FF.

From the extracted data, we observe that the average fanout changes depending on the region area and the number of FFs, but the average wirelength of driven nets at depths up to L is bounded. This implies that the tool increases N_g^l, which is the number of buffers at depth $l-1$, and the maximum depth of clock tree L, as A_{reg} increases, and as N_{ff} and T_{tran}^{max} decrease. Therefore, it is very important to accurately estimate both L and N_g^l at each clock depth. We estimate the clock wirelength using the following method. First, we determine the N_e^l, which is the number of cells driven by a buffer at depth $l-1$, and calculate N_g^l (i.e., N_g^{l+1}/N_e^l). Then, using the equation $\sqrt{A_{reg} \cdot N}$ from [22], we obtain Equation (1):

$$w^l = N_g^l \cdot \sqrt{A_{reg} \cdot \ N_e^l / N_g^l} \tag{1}$$

Extending Equation (1) to total clock wirelength, we obtain

$$\begin{aligned} W_{clk} &= \sum_{i=0}^{L} w^l = \sqrt{A_{reg}} \cdot (\sqrt{N_{ff}} + \sum_{i=0}^{L-1} \sqrt{N_e^i \cdot N_g^i}) \\ &= \sqrt{A_{reg} \cdot N_{ff}} \cdot (1 + \sum_{i=1}^{L} \sqrt{N_g^i}) = k \cdot \sqrt{A_{reg} \cdot N_{ff}} \end{aligned} \tag{2}$$

where k is a coefficient factor. However, Equation (2) does not account for the global clock tree that distributes the clock to the clusters. We extend Equation (2) as

$$W_{clk} = k_c \cdot \sqrt{A_{reg} \cdot N_{ff}} + k_g \cdot \sqrt{A_{reg} \cdot N_c} \tag{3}$$

where k_c and k_g are fitted coefficients for cluster and global clock trees. Note that both k_c and k_g are sensitive to A_{reg}, T_{tran}^{max}, N_{ff} and N_c.

Data path wirelength is proportional to the number of data wires and the cluster dimension.

$$W_{data} = k_0 \cdot N_{ff} \cdot \sqrt{A_{reg} / N_c} \tag{4}$$

For a large number of clusters, when the number of sinks per cluster is small, the data wirelength deviates from the above based on the placement tool runscript, since the sinks are not evenly distributed in the cluster. We do not consider such cases since they incur large power and area overheads due to over-clustering.

B. Clock and Data Buffer Area

Tellez and Sarrafzadeh [23] give a method to insert the minimum number of buffers under a given transition time (T_{tran}^{max}) constraint. They formulate a nonlinear constrained buffer insertion problem that minimizes the number of stages in a given rooted clock tree, such that for each stage, the stage rise time does not exceed T_{tran}^{max}. They then linearize this problem by using the concept of maximum capacitance (C_{max}). Maximum wirelength W_{max} is computed such that the rise time at the end of the wire is T_{tran}^{max}. Using the relationship $C_{max} = C_0 \cdot W_{max} + C_g$, where C_0 is the capacitance per unit length and C_g is the gate input capacitance, they conjecture that any buffer stage i with stage capacitance $C_i \leq C_{max}$ will have $T_{tran}^i \leq T_{tran}^{max}$. Based on this approach, we estimate the number of clock buffers (N_{cbuf}) as

$$N_{cbuf} = \frac{W_{clk} \cdot C_0 + N_{ff} \cdot C_g^{ff}}{C_{max} - C_g^{buf}} \tag{5}$$

where C_g^{buf} and C_g^{ff} are the input gate capacitance of a buffer and a flip-flop, respectively.

C_{max} is related to T_{tran}^{max} as follows, which is a consequence of the slew degradation discussed in [24]:

$$\begin{aligned} T_{tran}^{max} &= \sqrt{(T_0)^2 + (k_1 \cdot C_{max} \cdot R_{max})^2} \\ C_{max} &= k_2 \cdot \sqrt{(T_{tran}^{max})^2 - (T_0)^2} \end{aligned} \tag{6}$$

where T_0 is the transition time at the input to the line and R_{max} is the maximum resistance of W_{max}. The total buffer area (A_{clk}) is proportional to N_{cbuf}, so we have

$$A_{clk} = k_3 \cdot \frac{W_{clk} \cdot C_0 + N_{ff} \cdot C_g^{ff}}{\sqrt{(T_{tran}^{max})^2 - (T_0)^2}} \tag{7}$$

We again extend this expression to a clustered design, and add the clock buffers for the global tree and cluster tree:

$$A_{clk} = k_1 \cdot \frac{N_{ff} \cdot C_g^{ff} + W_{clus} \cdot C_0}{\sqrt{(T_{tran}^{max})^2 - (T_0)^2}} + k_2 \cdot \frac{N_c \cdot C_g^{buf} + W_{glob} \cdot C_0}{\sqrt{(T_{tran}^{max})^2 - (T_0)^2}} \tag{8}$$

where k_1 and k_2 are fitted coefficients. C_g^{ff} and C_g^{buf} are the flip-flop and buffer input gate capacitances, respectively, and C_0 is the wire capacitance per unit length.

The data buffer area is calculated as follows. Equation (7) can be used to calculate the number of buffers per data wire. Since some of the data wires could be quite small, we need to use a ceiling function to find an integral number of buffers per data wire as shown in Equation (9). Also, we need a minimum number of data buffers, n_{hold}, to ensure that hold time is not violated:

$$n_{dbuf}(w_{data}^i) = MAX(\left\lceil k_3 \cdot \frac{w_{data}^i \cdot C_0 + C_g^{ff}}{\sqrt{(T_{tran}^{max})^2 - (T_0)^2}} \right\rceil, n_{hold}) \tag{9}$$

where w_{data}^i is the wirelength of i^{th} data path. For each cluster, the data wires begin at the sinks that are distributed uniformly across the cluster and end at the 3DIO array at the center of the cluster, as shown in Figure 4.

Figure 4 shows how for a square of side d, and a pitch p of the placed sinks, there are concentric octagons with the same Manhattan wirelengths from the 3DIO. If we index each octagon i (starting from the center), then the number of sinks per octagon increases as $4 \cdot i$ until $i = \frac{d}{2p}$, beyond which the number of sinks per octagon decreases as $4 \cdot (\frac{d}{p} - i)$. We thus obtain the total number of data buffers in Equation (10), and consequently the data buffer area (A_{data}). Note

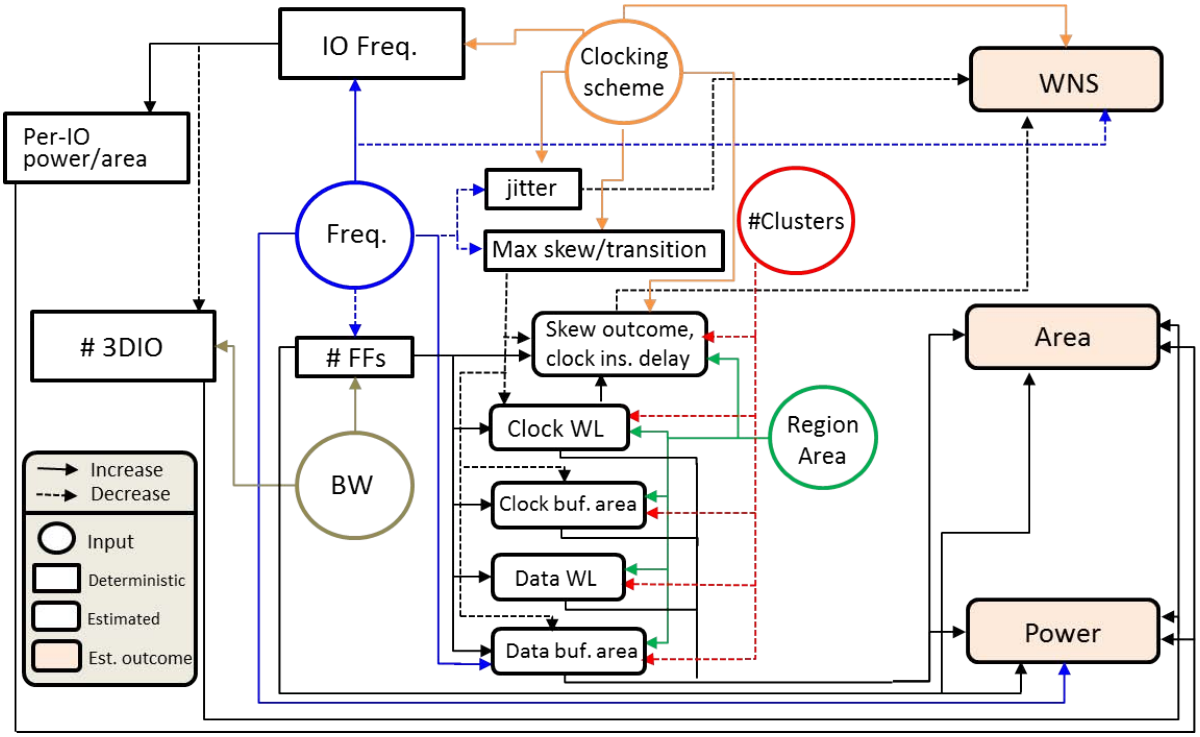

Fig. 3. 3DIO/CTS directed graph.

that we ignore the area of the 3DIO array and the other cases in which the placement of sinks does not exactly follow the octagon shape, since this equation is only to guide our metamodel for better estimation.

For the synchronous clocking case, the inter-die variation is large, which leads to a large number of hold buffers added in the design, and to a larger n_{hold} value for the 3DIO timing path.

$$A_{data} \propto N_{dbuf} = \sum_{i=1}^{\frac{d}{2p}} 4i \cdot n_{dbuf}(i \cdot p) + \sum_{i=\frac{d}{2p}+1}^{\frac{d}{p}-1} 4(\frac{d}{p} - i) \cdot n_{dbuf}(i \cdot p)$$
$$p = \sqrt{\frac{A_{reg}}{N_{ff}}}, \quad d = \sqrt{\frac{A_{reg}}{N_c}} \tag{10}$$

C. 3DIO Area and Power

The 3DIO power and area models are based on CACTI-IO [4]. The IO area equation is shown below in Equation (11). The fitting constants used are for a foundry 65nm technology to align with the CTS results.

$$A_{IO} = N_{IO} \cdot \left(A_0 + \frac{k_4}{min(R_{on}, 2 \cdot R_{TT1})}\right) + N_{IO} \cdot \left(\frac{1}{R_{on}}\right) \cdot k_5 \cdot (f_{IO}) + k_6 \cdot (f_{IO})^2 + k_7 \cdot (f_{IO})^3) \tag{11}$$

The IO power is a sum of the dynamic switching power, termination power and bias power as described in [4]. The load capacitance and termination frequency based threshold values have been nominally set based on a 3D interconnect. We use 750 MHz as the threshold for a pseudo-differential receiver consuming 100uA per IO, and 1500 MHz as the threshold for a termination consuming 1mA per IO. We also assume a static power overhead of 2.5mA for the PLL required in the asynchronous serial design per cluster. There are a number of energy-efficient serial IO designs that have been developed in the literature [25], [26], [27], [28], [29], [30], which could be applied to 3D interconnect as well, but we do not include all these schemes. Instead, we use a simple model for the asynchronous IO. The user using our flow can easily replace the 3DIO models with models from the library choices available and repeat the design space search that we illustrate below. Extending our model to include serial link design optimization [31], [32] may be interesting if the nature of these tradeoffs is to be studied in the context of the 3D clock clustering and 3DIO frequency. We plan to investigate this in the future.

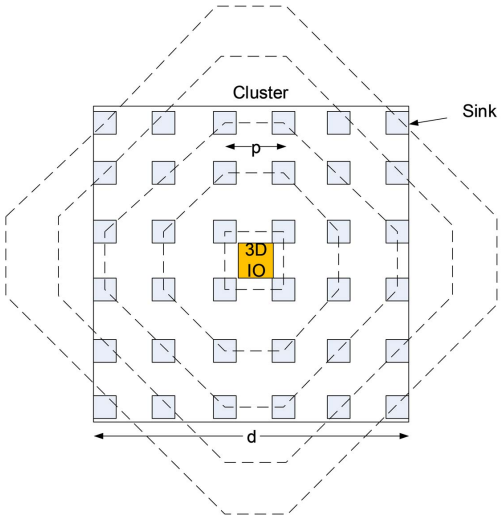

Fig. 4. Data wirelength distribution.

D. Total Power and Area

The total area is calculated as a sum of clock and data wirelength, buffer area, and 3DIO area, as shown in Equation (12). We treat

wire area and buffer area with equal weight for the purpose of our design space study, which might lead to large area values relative to the region area of the design, since multiple metal layers exist. One may wish to weight these unequally, e.g., if buffer area is a bigger concern than wire area, or vice-versa. The total power for the wire and buffers is proportional to their area (capacitance) and frequency, but also includes leakage and any static power which is independent of frequency but proportional to the number of buffers (see Equation (13)).

$$A_{total} = k_8 \cdot W_{clk} + k_9 \cdot W_{data} + A_{clk} + A_{data} + A_{IO} \tag{12}$$

$$\begin{aligned} P_{total} = (k_{10} \cdot W_{clk} + k_{11} \cdot W_{data} + k_{12} \cdot A_{clk} + \\ k_{13} \cdot A_{data}) \cdot F_{clk} + k_{14} \cdot (A_{clk} + A_{data}) + P_{IO} \end{aligned} \tag{13}$$

E. WNS and F_{max} for On-die and 3DIO

WNS is calculated for two paths, the first being the on-die path from launch in the cluster to capture in the 3DIO, and the second being the 3DIO die to die path.

For the on-die worst-case negative slack, all three synchronization schemes follow a synchronous full-cycle reg-to-reg timing, with the setup (WNS_{setup}) and hold (WNS_{hold}) slacks described as follows:

$$\begin{aligned} WNS_{setup} &= T_{per} - T_{data} - T_{hold\_fix} - T_{skew} - T_{jit\_setup} - T_{setup} \\ WNS_{hold} &= T_{hold\_fix} - T_{skew} - T_{jit\_hold} - T_{hold} \end{aligned} \tag{14}$$

where T_{per} is the clock period, T_{data} is the delay of the data as shown in Figure 2(a), T_{hold_fix} is the inserted delay to fix hold time, T_{skew} is the skew in the clock tree, T_{jit_setup} is the clock jitter, T_{jit_hold} is the 0-cycle uncertainty, T_{setup} and T_{hold} are the setup and hold times of the FF.

The worst negative slacks for the IO (WNS_{IO_setup}, WNS_{IO_hold}) for the three synchronization schemes are as follows.

Source-synchronous:

$$\begin{aligned} WNS_{IO\_setup} &= T_{per}/2 - T_{hold\_fix} - T_{skew} - T_{jit\_setup} - T_{setup} \\ WNS_{IO\_hold} &= T_{hold\_fix} - T_{skew} - T_{jit\_hold} - T_{hold} \end{aligned} \tag{15}$$

Synchronous:

$$\begin{aligned} WNS_{IO\_setup} &= T_{per}/2 - T_{data} - T_{hold\_fix} - \\ &\qquad T_{skew\_inter\_die} - T_{jit\_setup} - T_{setup} \\ WNS_{IO\_hold} &= T_{hold\_fix} - T_{skew\_inter\_die} - T_{jit\_hold} - T_{hold} \end{aligned} \tag{16}$$

Asynchronous:

$$\begin{aligned} WNS_{IO\_setup} &= T_{per}/8 - T_{jit\_setup} - T_{setup} \\ WNS_{IO\_hold} &= T_{hold\_fix} - T_{jit\_hold} - T_{hold} \\ T_{jit\_hold} &\simeq 0 \end{aligned} \tag{17}$$

For all three schemes, the skew and jitter are described as below, where T_{clk} is the clock insertion delay of the cluster clock. The fitted coefficients k_{15} to k_{19} are different for each clocking scheme.

$$\begin{aligned} T_{skew} &= k_{15} \cdot T_{clk} \\ T_{jit\_setup} &= k_{16} \cdot T_{clk} + k_{17} \cdot T_{per} \\ T_{clk} &= k_{18} \cdot A_{reg}/N_{clus} + k_{19} \cdot \frac{W_{clk} \cdot C_0 + N_{ff} \cdot C_g^{ff}}{\sqrt{(T_{tran}^{max})^2 - (T_0)^2}} \end{aligned} \tag{18}$$

The skew for the inter-die synchronous case is much larger due to inter-die process variation (much larger k_{15}). T_{data} also follows the same form as the T_{clk} shown above. As described in Section II, the source-synchronous case is not exposed to inter-die variation, and also does not have any data-delay eating into the setup slack due to the balance delay T_b shown in Figure 2(b).

F_{max} is calculated from the smallest T_{per} that still results in a positive WNS. F_{max} provides a better way to search for a valid design point and is also less prone to large % errors, unlike a WNS figure of merit when WNS is close to 0. For the asynchronous 3DIO, since the IO timing budget is mainly dominated by jitter and outside the scope of the timing flow described in Section IV, we assume 8000 Mbps as the maximum data rate. This means that the cluster frequency is limited to 1 GHz (due to 1:8 serialization) for the asynchronous case.

IV. P&R and Timing Flow

In this section, we describe different 3D P&R flows according to the different choice of clock synchronization schemes. We use a 65nm TSMC library with our P&R flows which are realized using Synopsys IC Compiler I-2013.12-SP1. Before describing the details of our flows, we clarify the following assumptions in our design:

- There are three kinds of launch-capture paths: (i) the on-die paths from the launch to the capture FFs on bottom die (H_0); (ii) the paths going through the 3DIOs that are from the FFs on the bottom die to the FFs on the top die (H_c); and (iii) the on-die paths from launch to capture FFs on the top die (H_1). The H_0 and H_1 paths do not go back and forth between bottom and top die.
- 3DIOs are placed in the center of each cluster in a square array with 30 μm pitch.[1]
- Launch (capture) FFs of H_0 (H_1) on the bottom (top) die are uniformly distributed within a given cluster region. Pitch between any two neighboring uniformly distributed FFs is $\sqrt{\frac{A_{reg}}{N_{ff}}}$. Capture (launch) FFs of H_0 (H_1) on the bottom (top) die are placed immediately next to the 3DIOs. We assume that the rest of the logic, which is not included in the database, does not block such a placement assumption.
- H_c are unidirectional paths from bottom die to top die and only have IO circuits which are (i) double data rate (DDR) for the synchronous and source-synchronous schemes; and (ii) 1:8 serializer, 8:1 deserializer for the asynchronous scheme. These circuits are placed right next to the 3DIOs.
- In timing analysis, we use a 7.5% clock uncertainty that includes clock jitter and OCV margin.
- For timing constraints, we use 5%, 75% and 20% of the clock period as max transition time, max clock insertion delay and max clock skew, respectively.

A. Synchronous

In the synchronous scheme, the clusters in both dies share a common cluster clock tree that is balanced to endpoints in both dies. Since the commercial tool cannot synthesize the clock tree concurrently considering both dies, we use the flow shown in Figure 5. First, we generate a gate-level netlist and timing constraints file (i.e., SDC file) for a given BW, A_{reg}, N_c and f. We then place FFs, DDR circuits and 3DIOs based on our assumptions described above. To balance the clock tree on both dies, we synthesize the cluster clock tree on the top die, and then extract the maximum clock insertion delay T_{clk1} of the clock tree as shown in Figure 2(a). Next, we annotate the delay to the clock 3DIO on the bottom die so that the commercial tool can consider the delay during the cluster clock tree synthesis of the bottom die. After CTS and routing on the bottom die, we propagate the T_{data} delays for the routing on the top die. Note that if the commercial tool performs the routing without consideration of T_{data}, the tool will insert many redundant hold buffers to compensate the clock insertion delay T_{clk1} in addition to the jitter T_{jit_hold} and FF hold time T_{hold}.

For timing analysis, we extract the netlist and parasitics (SPEF) of both the bottom and top dies, and merge the netlists and SPEFs as one

[1]We have sampled the sensitivity of the P&R results to aspect ratio of the region area, non-square cluster shape, 3DIO pitch and technology. We see that the trends in our data are not significantly affected by these assumptions. This being said, we have not included these parameters in a full-factorial DOE.

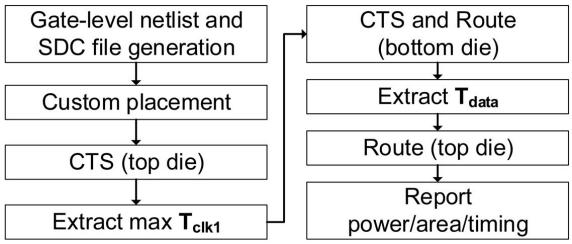

Fig. 5. Flow for synchronous clocking scheme.

netlist and SPEF file. We then use Synopsys PrimeTime to analyze setup and hold WNS. In timing analysis, we use best corner (*BC*) and worst corner (*WC*) for inter-die variation, and assign the same corner for paths that are on the same die. There are four combinations (i.e., *BC-BC*, *BC-WC*, *WC-BC*, *WC-WC*) and we report the worst setup and hold WNS out of the four combinations. We note that this is different from conventional timing analysis, which would analyze setup (hold) timing by assigning the worst (best) corner to the launch path and the best (worst) corner to the capture path, in consideration of intra-die variation. Here, we assign the same corner to the paths on the same die no matter whether the paths are launch or capture paths.

B. Source-Synchronous

The key feature of the source-synchronous scheme is to forward the clock to the clock 3DIOs which match the delays from the clock source to data 3DIOs on the bottom die. Figure 2(b) shows an example launch-to-capture path from bottom die to top die. Since the launch delays (i.e., T_{data} + T_b) up to data 3DIOs including balance delays, and the delays (i.e., T_{clk0} + T_{clk1}) up to clock pins of capture FFs, are well balanced, the inter-die variation is negligible. Note that we take into account 3DIO interconnect variation and interconnect loading mismatch by adding 5% margin to the balance delay (i.e., $T_b = 1.05 \times T_{clk1}$). However, special IO circuitry is required to insert the balance delay (T_b); the additional IO circuits increase total power and area compared to the synchronous scheme.

Figure 6 shows our flow for the source-synchronous scheme. The first two steps (i.e., generation of the gate-level netlist, and custom placement) are the same as for the synchronous scheme. After custom placement on the bottom and top die, we synthesize the clock tree and route on the bottom die, and separately synthesize the clock tree only for the top die. We then extract T_b (i.e., T_{clk1}) for each capture FF and annotate the delays to the corresponding data 3DIOs. These annotated delays guide the commercial tool to insert proper T_{hold_fix} only considering T_{jit_hold} and T_{hold}.

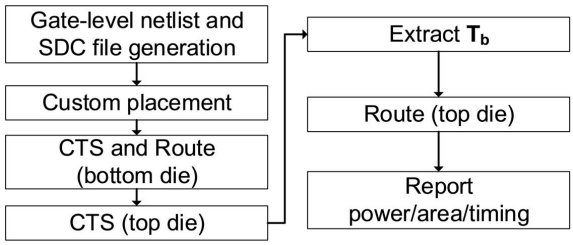

Fig. 6. Flow for source-synchronous clocking scheme.

Timing analysis is performed in the same way as for the synchronous scheme.

C. Asynchronous

In the asynchronous scheme, the cluster clock is not propagated from bottom to top die, and separate PLLs for the IO circuits exist. There is no inter-die variation or clock skew. Therefore, we can run the traditional 2D flow on both dies separately. The only difference in the gate-level netlist is that the asynchronous scheme uses serializer and deserializer, instead of DDR module.

V. Results

We now describe the DOE used to generate modeling data, the model-fitting results, and the design space visualization.

A. DOE

We construct a DOE where we vary the bandwidth (10-200 GB/s), region area (25-100 mm^2), 3DIO clock frequency (ranges based on the clocking scheme: synchronous (100-2000 Mbps), source-synchronous (1500-4000 Mbps) and asynchronous (3500-8000 Mbps) and number of clusters (1-25). We have collected data for over 256 design implementations for each of the three clock synchronization schemes. We have chosen these ranges based on typical 3D bandwidth requirement ranges [5] and typical speeds based on synchronization schemes [6].

We execute our DOE for the synchronous, source-synchronous and asynchronous schemes by running the P&R and timing flow described in Section IV. Since our P&R flow makes the various assumptions described in Section IV, we limit our DOE study to these design assumptions. We then fit and validate our clock tree and data path models based on the data from the DOE by logging all intermediate nodes of the directed graph in Figure 3.

B. Model Fitting

We use ANN models to fit the analytical models, as described in Section III. We make multiple runs with different training, validation and test data sets for improved generality and robustness of the resulting models.

The models for the internal nodes of the directed graph (clock wirelength, data wirelength, clock buffer area, data buffer area) fit within +/-20% error. Shown in Figures 7 and 8 are the distributions of % model error for total on-die area and power across the three synchronization schemes for all cases in the DOE.

The % errors of area models are 0.52/-17.8/19.76 (mean/min/max) for the synchronous scheme, -0.189/-11.28/8.56 for the source-synchronous scheme, and -0.08/-9.83/7.32 for the asynchronous scheme. Also, the % errors of power models are 0.099/-13.4/13.39 for the synchronous scheme, 0.018/-10.4/9.75 for the source-synchronous scheme, and -0.008/ -16.65/13.22 for the asynchronous scheme. In general, the models fit within +/-15% accuracy across a large range of bandwidth, region area, 3DIO clock frequency and number of clusters for all three synchronization schemes.

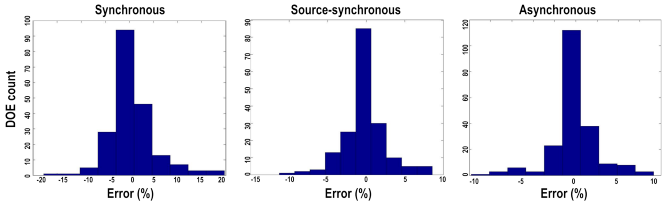

Fig. 7. Area model error %.

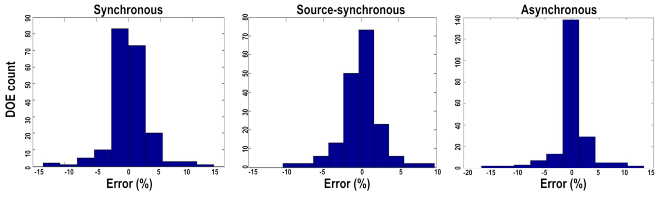

Fig. 8. Power model error %.

The F_{max} error % is shown in Figure 9. The mean/min/max % errors of F_{max} models are -0.1/-14.78/13.91 for the synchronous

scheme, 0.1/-10.02/10.15 for the source-synchronous scheme, and -0.1877/-9.47/ 12.3 for the asynchronous scheme. We use the F_{max} model to evaluate which design points meet timing, and hence are valid entries in the design space for power and area optimization. As indicated in Section IV, the timing for the on-die case is quite similar for the three synchronization schemes, but the 3DIO synchronization is very different: the synchronous scheme has the lowest F_{max}, followed by the source-synchronous scheme and then the asynchronous scheme, which achieves the highest 3DIO frequency.

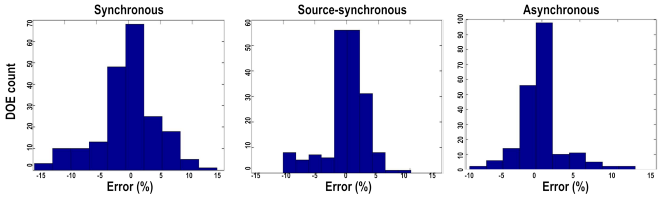

Fig. 9. F_{max} model error %.

As noted above, use 3DIO power and area analytical models from CACTI-IO [4]; and CACTI-IO model fitting results are given in [4]. We combine the CACTI-IO models for the 3DIO power and area with the ANN models. Given these models for power, area and timing, we may visualize the design space, as we discuss next.

C. Design Space

For an optimal power solution given an upper bound on area, we want to pick the lowest 3DIO frequency that meets the area constraint (as frequency goes lower the 3DIO area, sink area, and wire area all go up). This is because lower-frequency and wider buses are lower power for both the 3DIO and the clock tree. Once we have the lowest 3DIO frequency that meets this area constraint, we pick the 3DIO topologies that work at this frequency and search for the one with the lowest power. As the area constraint becomes tighter, the lowest 3DIO frequency that meets this area constraint becomes higher and requires us to move toward asynchronous/highly-clustered solutions. The caveat to this is that once the 3DIO frequency becomes high enough to require termination, the power efficiency of the 3DIO begins to improve as we scale frequency higher. This makes for an interesting tradeoff as we get to the asynchronous synchronization scheme. Further, the asynchronous synchronization scheme affords the unique advantage of having a slower cluster clock frequency for an equivalent 3DIO frequency due to the 1:8 serialization, so the CTS power can be quite low at relatively high 3DIO frequencies.

The design space is depicted in Figure 10. We use axes of max power constraint versus max area constraint, and show iso-bandwidth lines. We observe that the iso-bandwidth contour lines hit a vertical and horizontal *wall*. That is, for a given bandwidth, there is a minimum power and minimum area achievable across all synchronization schemes, number of clusters and cluster clock frequency.

Figures 11, 12 and 13 show how these three variables change as we move along the iso-bandwidth lines. Figure 11 shows the synchronization scheme, Figure 12 shows the number of clusters and Figure 13 shows the cluster clock frequency. The black dotted lines in Figures 10, 12 and 13 are the overlay of the three synchronization schemes shown in Figure 11.

We observe that the asynchronous scheme is area-efficient while the synchronous scheme is power-efficient. The source-synchronous scheme provides a valuable tradeoff between power and area along the knee of the iso-bandwidth curve. The interesting tradeoffs between the schemes occur along these knee points as we change the power/area constraint tradeoffs. The synchronous scheme achieves lower power at lower frequencies since there is no overhead associated with balancing delays (of the source-synchronous scheme) or with serialization (of the asynchronous scheme). The asynchronous

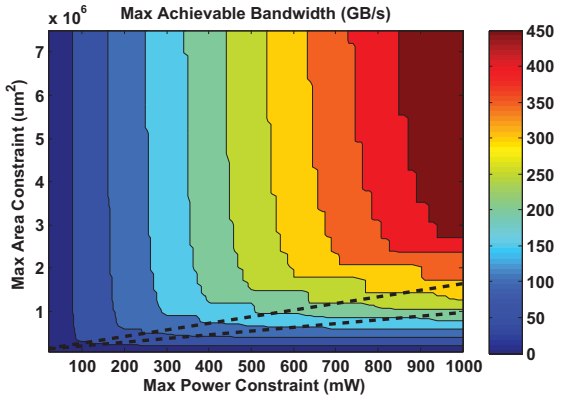

Fig. 10. Maximum achievable bandwidth for given power and area constraints.

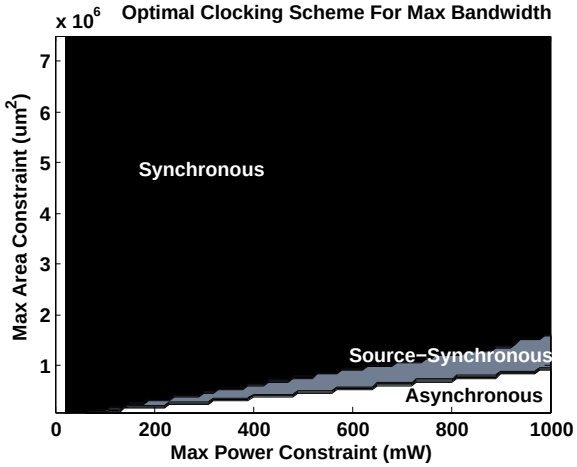

Fig. 11. Guidance for synchronization scheme for given power and area constraints.

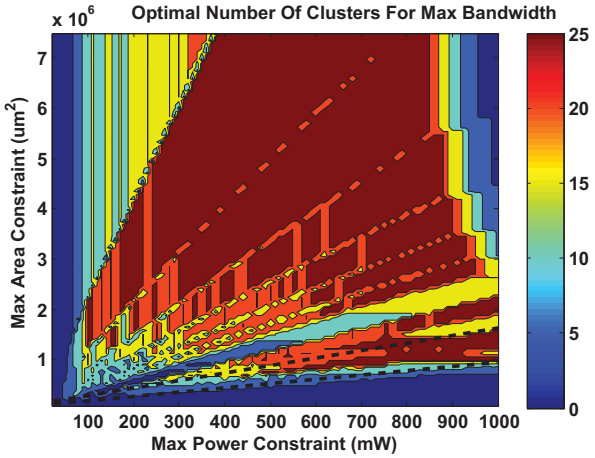

Fig. 12. Optimal number of clusters for given power and area constraints.

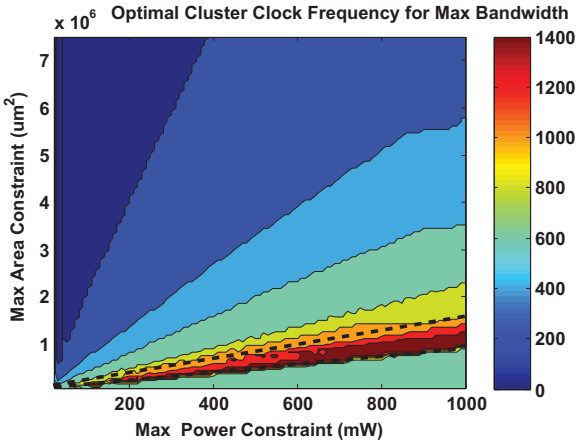

Fig. 13. Cluster clock frequency based on power and area constraints.

scheme achieves a lower area as it can operate at a higher frequency and needs fewer 3DIO.

We also observe that the optimal cluster clock frequency and number of clusters – for a given area constraint – both increase as we increase bandwidth (or the power constraint). However, the trend for number of clusters is not monotonic when we change synchronization schemes, and is also sensitive to the edge of the search hypercube (e.g., the largest bandwidth in the search window is 450 GB/s, so when this is reached, the bandwidth lines larger than 450 GB/s are limited to 450 GB/s, and the number of clusters stops following the trend as shown in the right top of Figure 12).

VI. Summary

We have presented a power, area and timing model for 3DIO and CTS that includes clustering and three different clock synchronization schemes (synchronous, source-synchronous, asynchronous). The model combines analytical closed-form expressions with metamodel-based fitting to achieve within 10% error for power, area and timing across a large range of bandwidths, region areas, numbers of clusters and 3DIO frequencies. Such a model enables us to visualize the design space and support architectural optimization with respect to choice of synchronization scheme, number of clusters and 3DIO frequency for given power and area constraints, as well as bandwidth targets. We believe that our modeling methodology will enable architects to study and optimize such a design space.

Our results show an interesting structure of the design space for the synchronization scheme, number of clusters and 3DIO frequency, wherein the area and power constraints on the design clearly affect the optimal design choices. Generally, the synchronous scheme is more power-efficient but has poor area efficiency, while the asynchronous scheme is more area-efficient and reduces the number of IO but has poor power efficiency. The source-synchronous scheme provides a good design point along the knee of the power-area tradeoff for a given bandwidth. Further optimization of the asynchronous serial IO could extend the asynchronous design space by lowering the power. Our ongoing work extends our study to include such schemes, as well as 2.5D (interposer-based) designs.

References

[1] T.-Y. Kim and T. Kim, "Bounded Skew Clock Routing for 3D Stacked IC Designs: Enabling Trade-offs Between Power and Clock Skew", *Proc. International Green Computing Conference*, 2010, pp. 525-532.

[2] X. Zhao, J. Minz and S. K. Lim, "Low-Power and Reliable Clock Network Design for Through-Silicon Via (TSV) Based 3D ICs", *IEEE Trans. on CPMT* 1(2) (2011), pp. 247-259.

[3] A. B. Kahng, B. Lin and S. Nath, "Explicit Modeling of Control and Data for Improved NoC Router Estimation", *Proc. DAC*, 2012, pp. 392-397.

[4] N. P. Jouppi, A. B. Kahng, N. Muralimanohar and V. Srinivas, "CACTI-IO: CACTI With Off-Chip Power-Area-Timing Models," *Proc. ICCAD*, 2012, pp. 294-301.

[5] JEDEC Wide IO Specification JESD229.

[6] W. Dally and J. Poulton, *Digital Systems Engineering*, Cambridge University Press, 1998.

[7] M. M. Navidi and G.-S. Byun, "Comparative Analysis of Clock Distribution Networks for TSV-based 3D IC Designs", *Proc. ISQED*, 2014, pp. 184-188.

[8] X. Dong and Y. Xie, "System-Level Cost Analysis and Design Exploration for Three-Dimensional Integrated Circuits (3D ICs)", *Proc. ASP-DAC*, 2009, pp. 234-241.

[9] X. Li, W. Liu, H. Du, Y. Wang, Y. Ma and H. Yang, "Whitespace-Aware TSV Arrangement in 3D Clock Tree Synthesis", *Proc. ISVLSI*, 2013, pp. 115-120.

[10] M. Mondal, A. J. Ricketts, S. Kirolos, T. Ragheb, G. Link, N. Vijaykrishnan and Y. Massoud, "Thermally Robust Clocking Schemes for 3D Integrated Circuits", *Proc. DATE*, 2007, pp. 1-6.

[11] M. Saint-Laurent and M. Swaminathan, "Impact of Power-Supply Noise on Timing in High-Frequency Microprocessors" *IEEE Trans. on Advanced Packaging* 27(1) (2004), pp. 135-144.

[12] C. Chu and D. F. Wong, "Closed Form Solution to Simultaneous Buffer Insertion/Sizing and Wire Sizing" *ACM Trans. on Design Automation of Electronic Systems* 6(3) (2001), pp. 343-371.

[13] R. Li, D. Zhou, J. Liu and X. Zeng, "Power-Optimal Simultaneous Buffer Insertion/Sizing and Wire Sizing", *Proc. ICCAD*, 2003, pp. 581-586.

[14] X. Liu, Y. Peng and M. C. Papaefthymiou, "Practical Repeater Insertion For Low Power: What Repeater Library Do We Need?", *Proc. DAC*, 2004, pp. 30-35.

[15] X. Wu, W. Zhao, M. Nakamoto, C. Nimmagadda, D. Lisk, S. Gu, R. Radojcic, M. Nowak and Y. Xie, "Electrical Characterization for Intertier Connections and Timing Analysis for 3-D ICs" *IEEE Trans. on VLSI Systems* 20(1) (2012), pp. 186-191.

[16] J. Jang, O. Franza and W. Burleson, "Compact Expressions for Period Jitter of Global Binary Clock Trees", *Proc. IEEE In EPEP*, 2008, pp. 47-50.

[17] D. Kim, J. Kim, J. Choi, J. S. Park, J. Kim, H. Lee, J. Lee and K. Park, "Distributed Multi TSV 3D Clock Distribution Network in TSV-Based 3D IC", *Proc. EPEPS*, 2011, pp. 87-90.

[18] W. L, H. Du, Y. Wang, Y. Ma, Y. Xie, J. Quan and H. Yang, "TSV-Aware Topology Generation for 3D Clock Tree Synthesis", *Proc. ISQED*, 2013, pp. 300-307.

[19] Y. Tsai, Y. Xie, N. Vijaykrishnan and M. J. Irwin, "Three-Dimensional Cache Design Exploration Using 3DCacti", *Proc. ICCD*, 2005, pp. 519-524.

[20] Y. Xie, G. H. Loh, B. Black and K. Bernstein, "Design Space Exploration for 3D Architectures" *ACM J. on Emerging Technologies in Computing Systems* 2(2) (2006), pp. 65-103.

[21] T. L. Snyder and J. M. Steele, "A priori bounds on the euclidean traveling salesman," *SIAM J. Computing* 24(3) 1995, pp. 665-671.

[22] J. M. Steele and T. L. Snyder, "Worst-case growth rates of some classical problems of combinatorial optimization," *SIAM J. Computing* 18(2) (1989), pp. 278-287.

[23] G. E. Tellez and M. Sarrafzadeh, "Minimal Buffer Insertion in Clock Trees with Skew and Slew Rate Constraints," *IEEE TCAD* 16(4) (1997), pp. 333-342.

[24] C. V. Kashyap, C. J. Alpert, F. Liu and A. Devgan, "Closed-Form Expressions for Extending Step Delay and Slew Metrics to Ramp Inputs for RC Trees," *IEEE TCAD* 23(4) (2004), pp. 509-516.

[25] H. Lee et al., "A 16 Gb/s/Link, 64 GB/s Bidirectional Asymmetric Memory Interface," *IEEE JSSC* 44(4) (2009), pp. 1235-1247.

[26] J. Poulton, R. Palmer, A. M. Fuller, T. Greer, J. Eyles, W. J. Dally and M. Horowitz, "A 14-mW 6.25-Gb/s Transceiver in 90-nm CMOS," *IEEE JSSC* 42(12) (2007), pp. 2745-2757.

[27] F. O'Mahony et al., "A 47x10Gb/s 1.4mW/(Gb/s) Parallel Interface in 45nm CMOS," *Proc. ISSCC*, 2010, pp. 156-158.

[28] R. Palmer, J. Poulton, A. Fuller, J. Chen and J. Zerbe, "Design Considerations for Low-Power High-Performance Mobile Logic and Memory Interfaces," *Proc. ASSCC*, 2008, pp. 205-208.

[29] J. Ellis, "Overcoming Obstacles for Closing Timing for DDR3-1600 and Beyond," *Denali MemCon*, 2010.

[30] A. Vaidyanath, "Challenges and Solutions for GHz DDR3 Memory Interface Design," *Denali MemCon*, 2010.

[31] V. Stojanovic and M. Horowitz, "Modeling and Analysis of High-Speed Links," *Proc. CICC*, 2003, pp. 589-594.

[32] R. Sredojevic and V. Stojanovic, "Optimization-Based Framework for Simultaneous Circuit-and-System Design-Space Exploration: A High-Speed Link Example," *Proc. ICCAD*, 2008, pp. 314-321.