We can observe how 2-SPP networks are significantly smaller in size than SOPs and SPPs (see for example the benchmarks *m4* and dist in Table II).

In summary, the experiments show that 2-SPP forms provide a very good compromise between the compact representation, the complexity of the minimization process, and testability. Besides being more efficient than the SOP regarding area, they are so far the only three-level forms that ensure full testability of the resulting circuit by construction.

VI. Conclusion

In this paper, we studied for the first time the testability of minimal 2-SPP and SPP networks for two static FMs, i.e., the SAFM and the CFM. For specific classes, i.e., 2 SPPs and SPPs minimal with respect to the number of literals in any variable ordering, a full testability has been proven for the SAFM, while for other classes, counter examples were provided. Networks that are not fully testable have been studied in order to improve their testability. 2-SPP and SPP networks minimal with respect to the number of pseudoproducts can be transformed into minimal fully testable forms.

The redundancy removal technique is a post-processing method, i.e., it is applied after the miminization of the network. Future work includes developing a synthesis approach that directly incorporates this technique during the minimization process.

Acknowledgment

The authors would like to thank the anonymous reviewers for their constructive comments that helped improve the quality of the paper.

References

[1] V. Ciriani, A. Bernasconi, and R. Drechsler, "Testability of SPP three-level logic networks," in *Proc. IFIP 12th Int. Conf. VLSI-SoC*, Darmstadt, Germany, 2003, pp. 331–336.

[2] O. Coudert, "Two-level logic minimization: An overview," *Integr. VLSI J.*, vol. 17, no. 2, pp. 97–140, Oct. 1994.

[3] T. Sasao, "AND–EXOR expressions and their optimization," in *Logic Synthesis and Optimization*, T. Sasao, Ed. Boston, MA: Kluwer, 1993.

[4] D. Debnath and T. Sasao, "Multiple–valued minimization to optimize PLAs with output EXOR gates," in *Proc. IEEE Int. Symp. Multiple-Valued Logic*, Freiburg, Germany, 1999, pp. 99–104.

[5] E. Dubrova, D. Miller, and J. Muzio, "AOXMIN-MV: A heuristic algorithm for AND–OR–XOR minimization," in *Proc. 4th Int. Workshop Applications Reed Muller Expansion Circuit Design*, Victoria, BC, Canada, 1999, pp. 37–54.

[6] D. Debnath and Z. Vransic, "A fast algorithm for OR–AND–OR synthesis," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 22, no. 9, pp. 1166–1176, Sep. 2003.

[7] V. Ciriani, "Three-level logic synthesis: Algebraic approach and minimization algorithms," Ph.D. dissertation, Dipartimento di Informatica, Univ. Pisa, Pisa, Italy, 2003.

[8] F. Luccio and L. Pagli, "On a new Boolean function with applications," *IEEE Trans. Comput.*, vol. 48, no. 3, pp. 296–310, Mar. 1999.

[9] R. Ishikawa, T. Igarashi, T. Hirayama, and K. Shimizu, "Pseudocube-based expressions to enhance testability," in *Proc. IEEE Asia-Pacific Conf. Circuits and Systems*, Singapore, 2002, vol. 2, pp. 305–310.

[10] R. Ishikawa, T. Hirayama, G. Koda, and K. Shimizu, "New three-level Boolean expression based on EXOR gates," *IEICE Trans. Inf. Syst.*, vol. E87-D, no. 5, pp. 1214–1222, 2004.

[11] V. Ciriani, "Synthesis of SPP three-level logic networks using affine spaces," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 22, no. 10, pp. 1310–1323, Oct. 2003.

[12] T. Williams and K. Parker, "Design for testability—A survey," *IEEE Trans. Comput.*, vol. C-31, no. 1, pp. 2–15, Jan. 1982.

[13] M. Breuer and A. Friedman, *Diagnosis & Reliable Design of Digital Systems*. Rockville, MD: Computer Science, 1976.

[14] M. Abramovici and M. Breuer, *Digital Systems Testing and Testable Design*. Piscataway, NJ: IEEE, 1994.

[15] A. Friedman, "Easily testable iterative systems," *IEEE Trans. Comput.*, vol. C-22, no. 12, pp. 1061–1064, Dec. 1973.

[16] V. Ciriani and A. Bernasconi, "2-SPP: A practical trade-off between SP and SPP synthesis," in *Proc. 5th IWSBP*, Freiberg, Germany, 2002, pp. 133–140.

[17] A. Bernasconi, V. Ciriani, F. Luccio, and L. Pagli, "Three-level logic minimization based on function regularities," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 22, no. 8, pp. 1005–1016, Aug. 2003.

[18] V. Ciriani, F. Luccio, and L. Pagli, "Synthesis of integer multipliers in sum of pseudoproducts form," *Integr. VLSI J.*, vol. 36, no. 3, pp. 103–118, Oct. 2003.

[19] M. Eggerstedt, N. Hendrich, and K. von der Heide, "Minimization of parity-checked fault-secure AND/EXOR networks," in *Proc. IFIP WG 10.2 Workshop Applications Reed-Muller Expansion Circuit Design*, Hamburg, Germany, 1993, pp. 142–146.

[20] J. Rose, A. El Gamal, and A. Sangiovanni-Vincentelli, "Architecture of field-programmable gate arrays," *Proc. IEEE*, vol. 81, no. 7, pp. 1013–1029, Jul. 1993.

[21] V. Ciriani, A. Bernasconi, and R. Drechsler, "Testability of SPP three-level logic networks in static fault models," Comput. Sci. Dept., Univ. Pisa, Pisa, Italy, Tech. Rep. TR-05-18, 2005.

[22] E. Sentovich, K. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. Stephan, R. Brayton, and A. Sangiovanni-Vincentelli, "SIS: A system for sequential circuit synthesis," Univ. California, Berkeley, Tech. Rep. UCB/ERL M92/41, 1992.

[23] S. Yang, "Logic synthesis and optimization benchmarks user guide version 3.0," Microelectron. Center North Carolina, Chapel Hill, NC, Tech. Rep., 1991.

Block-Level 3-D Global Routing With an Application to 3-D Packaging

Jacob Minz and Sung Kyu Lim

***Abstract*—Three-dimensional (3-D) packaging via system-on-a-package (SOP) has been recently proposed as an alternative solution to overcome the limitation of system-on-a-chip (SOC) and meet the rigorous requirements of today's mixed signal system integration. The true potential of SOP technology lies in its capability to integrate both active and passive components into a single high speed/density 3-D packaging substrate. The routing environment for 3-D SOP is more advanced than that of the conventional printed circuit board (PCB) or multichip module (MCM) technology—pins are located at all layers of the SOP packaging substrate rather than the topmost layer only, and various types of vias are available for layer-to-layer connections. The contribution of this work is to provide: 1) the formulation of the new block-level 3-D global routing problem under wire length, layer, crosstalk, and congestion minimization and 2) the first global router for 3-D SOP named 3PGR. This paper reviews various approaches for MCM routing algorithms and investigates their applicability to the SOP model. The related experimental results demonstrate the effectiveness of the algorithms.**

***Index Terms*—System-on-a-package, 3-D global routing, 3-D packaging.**

I. Introduction

The semiconductor industry is beginning to question the viability of system-on-a-chip (SOC) approach due to its low-yield and high-cost problem. Recently, three-dimensional (3-D) packaging via system-on-a-package (SOP) [1], [2] has been proposed as an alternative solution

Manuscript received October 9, 2004; revised April 24, 2005. This work was supported in part by the National Science Foundation under Contract CNS-0411149. This paper was recommended by Associate Editor M. D. F. Wang.

The authors are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: limsk@ece.gatech.edu).

Digital Object Identifier 10.1109/TCAD.2005.860952

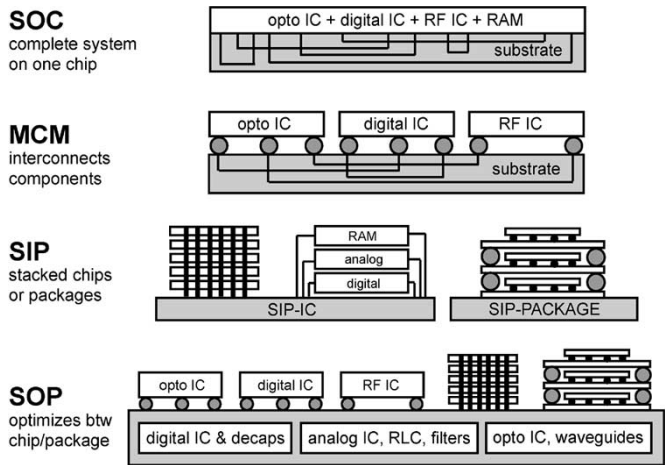

Fig. 1. Comparison among SOC, MCM, SIP, and SOP.

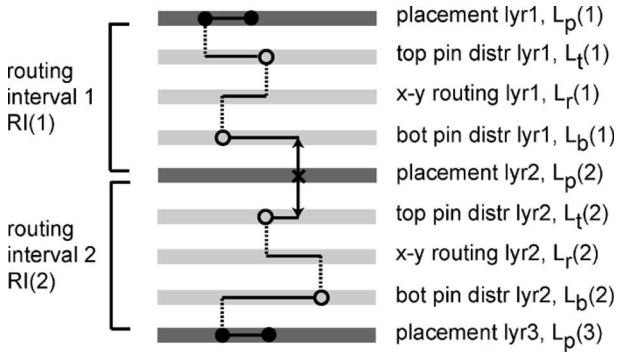

Fig. 2. Illustration of the layer structure and routing resource in SOP. The block and white dots, respectively, denote the original and redistributed pins. The "x" denotes a feed-through pin for an x-net to pass through a placement layer using a routing channel. The solid, dotted, and arrowed lines denote signal wires, vias, and feed-through vias, respectively.

to meet the rigorous requirements of today's mixed signal system integration. An illustration is shown in Fig. 1. The true potential of SOP technology lies in its capability to integrate both active and passive components into a single high speed/density 3-D packaging substrate. Each device layer is used to mount bare digital/analog dies (possibly using different technologies) and integrate embedded passive elements, and routing layers and vias are used to connect various elements. This provides a cost-effective and high-yield solution to system integration compared to SOC. In addition, 3-D packaging offers a significant saving in area, delay, and power compared to the conventional two-dimensional (2-D) packaging [printed circuit board (PCB) and multichip module (MCM)]. The layer-to-layer connection in 3-D SOP is more effectively done using various types of vias compared to wire-bonding-based or stacked 3-D system-in-a-package (SIP) [3]. Thus, innovative ideas on computer-aided design (CAD) tools for SOP technology are crucial to fully exploit the potential of this new emerging technology. However, there exist very few tools, if not none, that handle the complexity of automatic 3-D SOP layout generation. Some initial works recently published on physical design for 3-D SOP include [4]–[11].

Several MCM routing algorithms have been proposed in the literature [12]–[17]. Several works on MCM pin redistribution include [18], [19], and [20]. A notable difference between SOP and MCM routing lies in the fact that there exists multiple device layers in SOP, whereas in MCM there is only one device layer. Therefore, nets are now connecting pins located in all intermediate layers in SOP, and the blocks in each layer behave as obstacles. In MCM, however, all pins are located only at the top layer, and there exist no obstacles except for the wires themselves. This makes the SOP or 3-D package routing problem more general than MCM routing. Therefore, the existing MCM routers cannot be used directly for the design of SOP. In addition, recently developed physical design tools for 3-D ICs [21]–[35] are not applicable either since these tools target individual gates, whereas in 3-D packaging we place-and-route the blocks that represent chips and embedded passives. Therefore, our primary goal is to make the best use of placement and routing layers available while automatically generating 3-D package layout under various noise constraints. In this paper, we present 3PGR, the first global router for 3-D packaging. The contribution of this work is threefold.

1) We provide the formulation of the new block-level global routing problem for 3-D SOP under wire length, layer, crosstalk, and congestion minimization under various capacity constraints.
2) We formulate and design heuristics for the following new problems: a) SOP pin redistribution problem; b) SOP net distribution problem; and c) SOP channel assignment problem.
3) Our linear-time multiphase 3PGR algorithm efficiently achieves high-quality results.

The remainder of this paper is organized as follows. Section II presents the problem formulation and an overview of our 3PGR router. Sections III, IV, and V present our pin redistribution, layer assignment, and channel assignment algorithms, respectively. Section VI presents the experimental results. We then conclude in Section VII.

II. Problem Formulation

A. SOP Routing Resource

The layer structure in a multilayer SOP is illustrated in Fig. 2. The placement layers[1] contain the blocks (such as ICs, embedded passives, optoelectric components, etc.), which from the point of view of physical design are just rectangular blocks with pins along the boundary. The interval between two adjacent placement layers is called the routing interval. A routing interval contains a stack of routing layers sandwiched between pin distribution layers. These layers are actually $x-y$ routing layer pairs so that the rectilinear partial net topologies may be assigned to them. The pin distribution layers in each routing interval are used to evenly distribute pins from the nets that are assigned to this interval. Then, these evenly distributed pins are connected using the routing layer pairs. Each placement layer consists of a pair of $x-y$ routing layers, so routing is permitted. A feed-through via is used to connect two pin distribution layers from different routing intervals. Thus, the routing channels in each placement layer are used for two purposes: 1) accommodate feed-through vias and 2) perform local routing, where a limited number of intralayer connections are made.

In the SOP model, the nets are classified into two categories. The nets that have all their terminals in the same placement layer are called i-nets, while the ones having terminals in different placement layers are x-nets. The i-nets can be routed in a single routing interval or indeed within the placement layer itself. On the other hand, the x-nets may span more than one routing interval. Table I shows five different types of nets existing in SOP global routing along with their layer usage. Fig. 3 shows the corresponding illustration.

[1]We use placement layer and device layer interchangeably.

TABLE I
TYPE OF NETS AND LAYER USAGE FOR A 3-D SOP WITH k PLACEMENT LAYERS. THE LAYER INFORMATION IS DENOTED IN PARENTHESIS

type	pins	layers used
1	$p_1(1)$ and $p_2(1)$	$L_p(1)$, $L_t(1)$, $L_r(1)$
2	$p_1(K)$ and $p_2(K)$	$L_p(K)$, $L_b(K\text{-}1)$, $L_r(K\text{-}1)$
3	$p_1(i)$ and $p_2(i)$	$L_p(i)$, $L_t(i)$, $L_r(i)$ or $L_p(i)$, $L_b(i\text{-}1)$, $L_r(i\text{-}1)$
4	$p_1(i)$ and $p_2(i{+}1)$	$L_p(i)$, $L_t(i)$, $L_r(i)$, $L_b(i)$, $L_p(i{+}1)$
5	$p_1(i)$ and $p_2(i{+}2)$	$L_p(i)$, $L_t(i)$, $L_r(i)$, $L_b(i)$, $L_p(i{+}1)$ $L_t(i{+}1)$, $L_r(i{+}1)$, $L_b(i{+}1)$, $L_p(i{+}2)$

The routing and pin distribution layers in each routing interval are modeled with a standard $x \times y \times z$ 3-D grid, where each node represents a routing region and each x/y-direction edge represents each horizontal/vertical boundary among the regions. Each z-direction edge represents a group of vias each region can accommodate. Thus, all edges in this 3-D grid are associated with capacity: x and y edges for wire capacity, and z edges for via capacity. We use the Floor Connection Graph (FCG) [36], illustrated in Fig. 4, to model the placement layer, where each routing channel becomes an edge and each channel intersection point becomes a node. In addition, each soft block becomes a node, and pin assignment edges are added to this node to connect to all adjacent channels. This model allows us to determine which boundary to use for the pins with unknown location. Each channel edge is associated with: 1) via capacity for feed-through vias and 2) wire capacity for local routing. In addition, pin assignment edges have a pin capacity for each boundary of a soft block.

B. SOP Routing Problem

For each net n from a given netlist NL, let xt_n, wl_n, and via_n, respectively, denote the amount of crosstalk, wire length, and via associated with n. The wire length wl_n is the sum of Manhattan distance in x, y, and z directions, where the z-direction is the height of the associated vias.[2] Let $cl(n, m)$ denote the coupling length between n and m as illustrated in Fig. 5. We define xt_n as

$$xt_n = \sum_{m \in NL, m \neq n} \frac{cl(n,m)}{|z(n) - z(m)|}$$

where $z(n)$ denotes the routing layer that contains net n. For each net n, let $d_n(i)$ denote the Elmore delay [37] at sink i. Then, the maximum sink delay of net n, denoted d_n, is $\max\{d_n(i)|i \in n\}$. The performance of an SOP global routing is estimated by

$$D^{\max} = \max\{d_n | n \in NL\}.$$

For each routing interval i in a 3-D package with K placement layers, let $L_t(i)$, $L_r(i)$, and $L_b(i)$, respectively, denote the top pin distribution layer pairs, routing layer pairs, and bottom pin distribution layer pairs. There exist three kinds of connections in each routing interval i: top [connection between $L_p(i)$ and $L_t(i)$], middle [connection between $L_t(i)$ and $L_b(i)$], and bottom [connection between $L_b(i)$ and $L_p(i+1)$]. We use the routing layer pairs in $L_t(i)$, $L_r(i)$, and $L_b(i)$, respectively, for the top, middle, and bottom connections. We construct the routing grid, denoted $G(i)$, that contains all the distributed pins from $L_t(i)$ and $L_b(i)$ in an $m \times n$ 2-D grid. We use $G(i)$ to perform topology generation for various two-pin and multipin connections. The total layer used in an SOP global routing solution is given by

$$L^{tot} = \sum_{1 \leq i \leq K} \left(|L_t(i)| + |L_r(i)| + |L_b(i)|\right).$$

Section IV-A discusses how to compute $|L_r(i)|$ and Section V-B discusses how to compute $|L_t(i)|$ and $|L_b(i)|$.

Lastly, the formal definition of an SOP global routing problem is as follows. Given a 3-D placement and netlist, generate a routing topology for each net n, assign n to a set of routing layers, and assign all pins of n to legal locations. All conflicting nets are assigned to different routing layers while satisfying various wire/via capacity constraints. The objective is to minimize the cost function

$$\alpha L^{tot} + \beta D^{\max} + \sum_{n \in NL} (\gamma xt_n + \delta wl_n + \epsilon via_n).$$

We minimize $|L_r|$ during our layer assignment step and $|L_t| + |L_b|$ during our channel assignment step. $D^{\max}$ is the focus during our topology generation step. Wire length, via, and crosstalk minimization are addressed in all steps of our global router.

C. Overview of 3PGR Algorithm

Our 3PGR router is divided into the following five steps.

1) Pin redistribution. We first determine which set of i-nets and x-net segments is assigned to each routing interval. The pins from these nets are then evenly distributed in the top and bottom pin distribution layers.
2) Topology generation. Steiner trees are generated for all nets in each routing interval so that the performance of the routed design is optimized.
3) Layer assignment. The routed nets are assigned to a unique routing pair in the routing layer so that the total number of layers used is minimized.
4) Channel assignment. For each x-net, its location of feed-through via in the routing channel is determined. We also assign channels and finish the connections for the i-nets that are to be routed in each placement layer.
5) Local routing. We finish connections between the pins now located in the routing channels and the pins on the block boundaries. We also determine the location of pins from soft blocks.

For step 2), we use an existing RSA/G heuristic [38] to generate the net topologies.[3] In addition, we use the congestion-driven rip-up-and-reroute [39] for step 5). Therefore, the focus of this paper is to develop heuristics for steps 1), 3), and 4). Pin redistribution is performed while considering all routing intervals simultaneously. During the topology generation and layer assignment, we visit each routing interval sequentially from top to bottom. During the channel assignment and local routing, we visit placement layers sequentially from top to bottom.

III. SOP PIN REDISTRIBUTION

We first present an overview of our approach and the problem formulation of SOP pin redistribution. We then discuss the details of the three steps used in our algorithm, namely, coarse pin distribution (CPD), net distribution, and detailed pin distribution (DPD).

[2]We assume that the height of vias connecting two $x-y$ layers in a routing and pin distribution layer pair is of one grid, whereas the height of feed-through vias that penetrate a placement layer is of six grid.

[3]In our approach, we first route all nets using minimum shortest path arborescence. We then rip-up and reroute nontiming critical nets for congestion control. During this reroute stage, we construct weighted arborescence, where the weights denote the current routing resource usage.

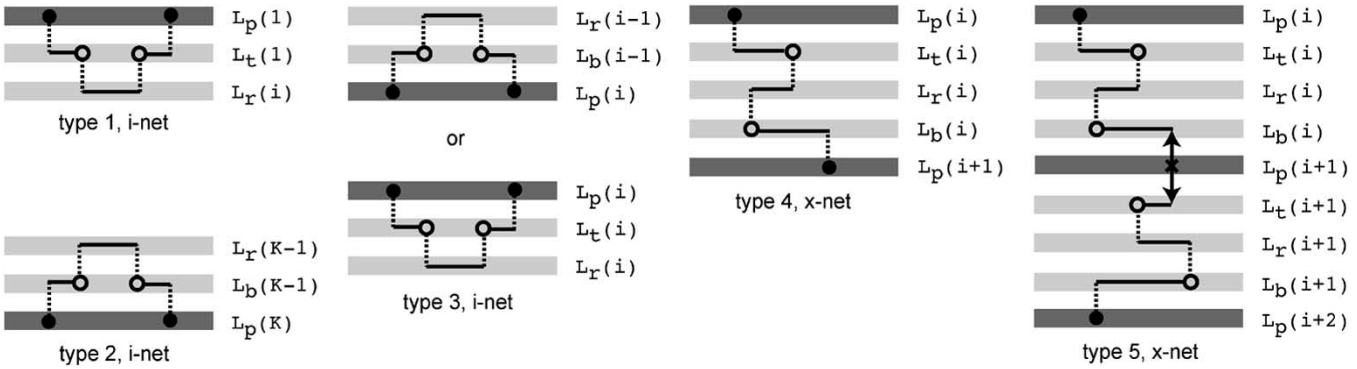

Fig. 3. Illustration of five different types of connections existing in SOP global routing. Note that MCM routing deals with only type 1 nets.

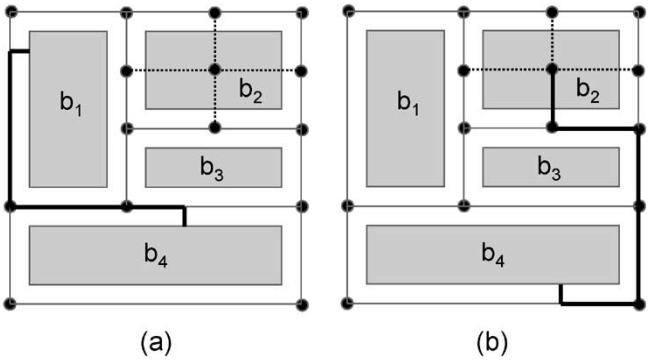

Fig. 4. Graph-based modeling of placement layer. (a) Connection between two hard blocks. (b) b_2 is a soft block. The new pin is located on the bottom boundary. Dotted lines denote π assignment edges.

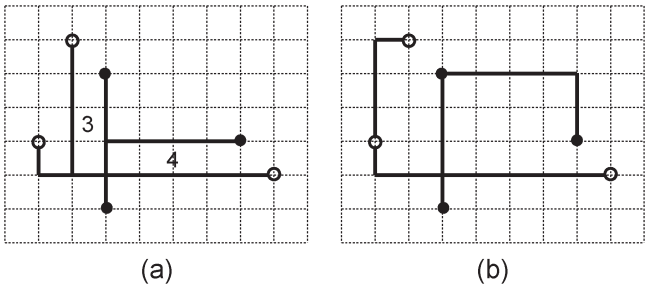

Fig. 5. Illustration of crosstalk computation between two Steiner trees. The numbers denote the coupling length. (a) Crosstalk is 7. (b) Crosstalk-free routing.

A. Overview of the Approach

During 3-D placement, we assume that pins are located at the center of the modules (= soft modules) or at the boundary of the modules (= hard module).[4] Thus, the pin location is highly localized and not evenly distributed. Since our plan is to use pin distribution layers and routing layers in combination to finish routing in each routing interval, one of the important steps is to evenly distribute pins in the pin distribution layer so that routing in the routing layers is done more evenly. This greatly helps reduce the number of routing layers used as well as crosstalk among nets. However, pin distribution cannot be done accurately without knowing which net is assigned to which routing interval. On the other hand, our net distribution needs to know the pin location for more accurate crosstalk measurement. Consequently, we need to iterate between pin distribution and net distribution until we converge to a good solution. We solve this issue with our three-stage effort: CPD, net distribution, and DPD.

1) CPD. We construct an $m \times n$ 2-D grid and evenly distribute the pins from all nets in all routing intervals in this single grid.
2) Net distribution. We assign a routing interval for each i-net to either above or below the placement layer it belongs to. The crosstalk computation is based on the CPD result.
3) DPD. We refine our pin distribution results for each routing interval based on the net distribution result. In addition, the pin location is legalized, i.e., each pin is assigned to a unique grid point in pin distribution layers.

B. Problem Formulation

The following is the set of inputs to the SOP pin redistribution problem: 1) a set of placement layers $L_p = \{L_p(1), L_p(2), \ldots, L_p(K)\}$; 2) a set of nets (i-nets and x-nets) $NL = \{n_1, n_2, \ldots, n_k\}$ that connect the pins in L_p; 3) a set of top pin distribution layers $L_t = \{L_t(1), L_t(2), \ldots, L_t(K)\}$; and 4) a set of bottom pin distribution layers $L_b = \{L_b(1), L_b(2), \ldots, L_b(K)\}$. A routing interval $RI(i)$ contains $L_p(i)$, $L_t(i)$, $L_r(i)$, $L_b(i)$, and $L_p(i+1)$. An illustration is shown in Fig. 2. Our goal is to determine 1) which routing interval(s) each i-net and x-net belongs to and 2) a one-to-one mapping from the pins in the placement layers to the pins in the redistribution layers. Each pin in L_p is assigned to a unique grid point in the $m \times n$ grid graph $G(i)$. Since each grid point represents a routing region in these pin distribution layers, each node/edge in $G(i)$ is associated with via/wire capacity. For a pin $p \in L_p$, let dw_p denote the wire length between the original and the new location (after pin redistribution). The objective of SOP pin redistribution is to minimize the following cost function under the via/wire capacity constraints, i.e.,

$$w_1 \sum_{p \in L_p} dw_p + \sum_{n \in NL} (w_2 wl_n + w_3 xt_n).$$

According to the five types of SOP nets shown in Fig. 3, we note that the routing interval assignment for some i-nets and all x-nets is straightforward: all i-nets from $L_p(1)$ are assigned to $RI(1)$ and all i-nets from $L_p(K)$ are assigned to $RI(K-1)$. In addition, each x-net that spans k routing intervals is decomposed into k segments and assigned to all intermediate routing intervals. Let $N^i = \{n_1^i, n_2^i, \ldots, n_k^i\}$ denote the set of movable i-nets that have pins from $L_p(i)$ for $1 < i < K$. Note that these movable nets can be assigned to either $L_b(i-1)$ or $L_t(i)$ while other nets are fixed into some intervals.

[4]A more general SOP pin redistribution may need to handle the distribution of pins from flip-chip dies that contain pins in a form of 2-D grid. This requires more sophisticated pin redistribution method to handle the pin/via congestion problem, which is out of the scope of this paper.

Coarse Pin Distribution
1: CP = $m \times n$ grid;
2: **for** (each pin $p \in NL$)
3: s = slot closest to p and under-utilized;
4: place p in s;
5: C = restricted multi-level clustering of pins;
6: hgt = height of C;
7: $B(hgt)$ = initial $m \times n$ placement at top level;
8: **for** ($i = hgt$ downto 0)
9: move clusters in $C(i)$ to optimize cost;
10: $B(i)$ = new $m \times n$ placement at level i;
11: project $B(i)$ to $B(i-1)$;
12: **return** $B(0)$;

Fig. 6. Pseudocode for CPD algorithm.

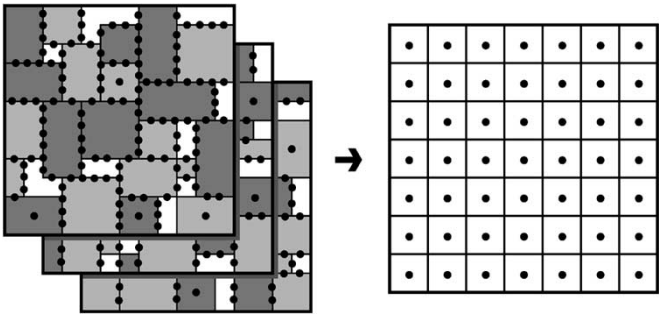

Fig. 7. Illustration of CPD. Pins along the external boundary are not shown for simplicity.

Thus, we formulate the SOP net distribution problem to decide which routing interval to use for the nets in N^i.[5]

C. Coarse Pin Distribution

A pseudocode for our CPD algorithm is shown in Fig. 6. First, we assign all pins in the placement layers to a nearby grid point in CP, an $m \times n$ 2-D grid, while trying to balance the number of pins assigned to each grid point (lines 1–4). An illustration is shown in Fig. 7. We impose pin capacity for each grid point so that the pins are evenly distributed in CP. Our approach is to visit the pins in random order and find the best grid for each pin. For each pin p, a grid point that is closest to the original location of p and has not violated the pin capacity constraint is chosen (line 3). After this process is finished, CP serves the starting point of our min-cut placement-based algorithm, where each grid point corresponds to a partition. We then iteratively improve the quality of this initial solution via move-based approach. We extend the multilevel min-cut-based global placement algorithm [41] for CPD. In [41], a recursive multilevel bipartitioning is used to divide the given netlist into an $m \times n$ grid while minimizing the number of inter-partition connections (= cutsize) as well as their estimated wire length. In our new heuristic algorithm, our cost function is based on 1) how far the new pin location is from the initial location; 2) total wire length; and 3) how evenly distributed the interpartition connections are.

[5]We attempted to solve the SOP net distribution problem using the existing K-way max-cut partitioning method [40]—we build the Net Interference Graph (NIG), where the nodes and edges, respectively, represent the nets and crosstalk between them. Then, the max-cut partitioning tries to separate nets with high crosstalk into different routing intervals. To our surprise, this approach had very little impact on crosstalk and produced results that were very close to a very simple heuristic: distribute the movable i-nets randomly. Our related experiments suggest that this is due to the small number of movable i-nets existing in our benchmarks (less than 10%). However, this does not mean that the SOP problem is insignificant. We believe that i-net distribution for crosstalk minimization will play an important role if the number of movable i-nets is huge.

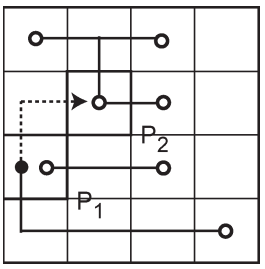

Fig. 8. Illustration of the gain computation for CPD. A net n indicated by the black node is moved from P_1 to P_2. Then, $g_d(n) = -2$ (two units away from the original location), $g_w = 0$ (wire length did not change), and $g_b = -1$ [$deg(P_2) = 3$ becomes the maximum-degree partition].

For each pin p, we define the displacement gain, denoted $g_d(p)$, to represent how much distance between the original and the new location is reduced if p is moved to another partition. We define the wire length gain, denoted $g_w(p)$, to represent how much the length of the nets that contain p (estimated by the half-perimeter of the bounding box) is reduced if p is moved to another partition. For a partition P, let $deg(P)$ denote the number of nets that have connections to P. Then, the cutsize balance factor is defined as

$$\max\{deg(P_i) - deg(P_j) | \forall P_i, P_j\}$$

that is, the difference between the maximum and the minimum degree among the partitions. We define the balance gain, denoted $g_b(p)$, to represent how much the cutsize balance factor is reduced. Our move-based multilevel min-cut partitioning algorithm performs cell move based on the combined gain function

$$g(p) = w_1 g_d(p) + w_2 g_w(p) + w_3 g_b(p).$$

Fig. 8 shows an illustration of the gain computation.

In our multilevel approach, we first perform the restricted multilevel clustering (line 5) that preserves the initial $m \times n$ placement result, where two pins that are in different partitions initially are not clustered together. At each level of the cluster hierarchy from top to bottom (line 8), we compute the combined gain $g(p)$ for each cluster and perform cluster moves. In order to compute the displacement and balance gain of a group of pins (= cluster), we add the individual displacement and balance gain of all pins in this cluster. When there is no gain at a certain level, we decompose the clusters into the next lower level and perform refinement. This process continues until we obtain a solution at the bottom level (line 12). Our initial partition computation takes $O(p \times m \times n)$ (lines 2–4), where p is the number of pins. Our multilevel clustering algorithm [42] (line 5) takes $O(p \log p)$, and the multilevel partitioning (line 8–11) takes $O(p)$. Therefore, the overall time complexity of our CPD algorithm is $O(p \times m \times n)$.

D. Detailed Pin Distribution

After CPD and net distribution are finished, we know which set of nets are assigned to each routing interval as well as their (evenly distributed) entry/exit points in pin distribution layers. However, the CPD is done based on the 2-D grid that merged all multiple placement layers into one. The even pin distribution in this 2-D grid offers a good enough reference point for net distribution. But, it does not consider even pin distribution in each individual routing interval. In addition, it is also possible that pin capacity for each routing region in each routing interval may be violated. Therefore, the goal of DPD is to address these problems in each routing interval so that the subsequent topology generation and layer assignment truly benefit from this even pin distribution. In addition, we use a grid large enough for each

Detailed Pin Distribution
1: **for** (each routing interval i)
2: $m_i = n_i = \lceil \sqrt{total\_pin(i)} \rceil$;
3: $DP[i] = m_i \times n_i$ grid;
4: **for** (each pin $p \in i$)
5: assign p to $DP[i]$ using CP result;
6: **for** (each routing interval i)
7: **for** (each net $n \in DP[i]$)
8: F_n = displacement force on n;
9: obtain new location in $DP[i]$ based on F_n;
10: L_i = sort nets in $DP[i]$ in lexicographic order;
11: split L_i and form columns in $DP[i]$;
12: assign pins in $DP[i]$ to clusters $Cl = \{cl_j\}$;
13: **for** (each cluster $cl_j \in CL$)
14: N_j = matching network for cl_j;
15: perform min-cost maximal matching on N_j;
16: obtain refined location for pins in $DP[i]$

Fig. 9. Pseudocode for SOP DPD algorithm.

routing interval to legalize pin location, i.e., each grid point contains only one pin. Since crosstalk minimization is addressed during prior steps, the major focus of the DPD step is on: 1) how far the new location is from the original location obtained from CPD and 2) the total wire length. In our hierarchical approach, we first perform a force-directed method to construct an initial distribution, which is used for the subsequent clustering phase. We then visit each cluster and perform network flow-based pin assignment.

A pseudocode for our DPD algorithm is shown in Fig. 9. Our force-directed heuristic algorithm encourages all pins from the same net to be placed closer to the center of mass while minimizing the distance between the old and the new pin location. The grid size for DPD for routing interval $i(= DP[i] = m_i \times n_i)$ is determined so that each pin can be assigned to a unique grid point. We compute $m_i = n_i = \lceil \sqrt{\text{total-pin}(i)} \rceil$ (lines 1–3). In addition, we project the CPD result $(= CP)$ to this new set of grids (line 5). Note that there still exists overlap among the pins in $DP[i]$ at this point even though DP is usually finer than CP. In order to remove this overlap, we apply an additional force that slightly pulls each pin toward the center of mass. For each pin p in a net n, the displacement force (line 8) for x-direction is defined as

$$F_x(p) = \frac{x(M_p) - x(p)}{\text{width}(n_p)}$$

where $x(M_p)$ denotes the x-coordinate of M, the center of mass of n, and width(n_p) denotes the width of the bounding box of n. We compute $F_y(p)$ using the y-coordinates. Note that $-1 \leq F_x(p)$, $F_y(p) \leq 1$. The vector $(F_x(p), F_y(p))$ is then added to (x_p, y_p) (line 9). This minor change on the original pin location helps to remove most of the overlap in $DP[i]$ while not increasing the wire length too much. We then sort the pins based on the lexicographic order of new locations and assign each pin starting from the topmost row in the leftmost column (lines 10–11). Fig. 10 shows an illustration. Due to its simplicity, this deterministic algorithm is quite efficient and effective in reducing the additional wire length required for pin distribution as well as the total wire length among all nets as shown in Section VI. The complexity of this algorithm is $O(p)$, where p denotes the total number of pins.

In order to achieve higher solution quality, the min-cost network flow is used to further reduce wire length and congestion. We first group the distributed pins in each routing interval into clusters based on their location (line 12, Fig. 9) and perform network flow-based matching for the pins in each cluster to find a new location (lines 13–16,

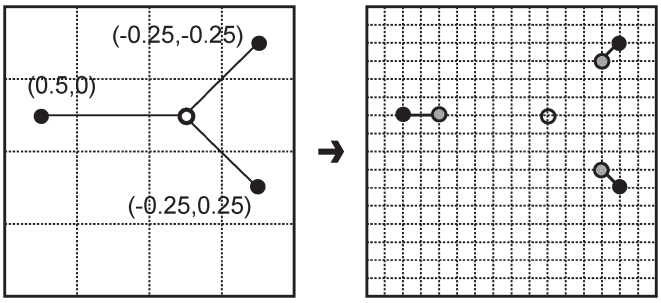

Fig. 10. Illustration of DPD algorithm. The black and gray nodes denote the old and new pin location, respectively, where the white node denotes the center of mass. The numbers denote the displacement force for each pin.

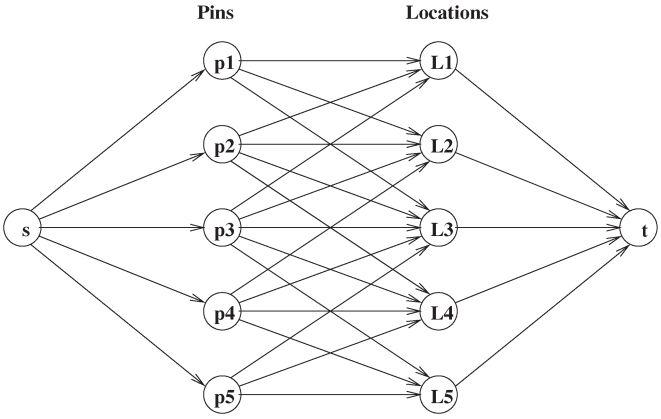

Fig. 11. Illustration of the pin-to-location flow network.

Fig. 9).[6] A matching network $N = (P, L, E, c)$ consists of a set of pins P, a set of locations L, a set of assignment edges $E = \{(u, v) : u \in P\ v \in L\}$, and a cost function $c : E \rightarrow R$. For each pin p_i, a set of assignments is formed by choosing neighboring locations l_j. The cost of the assignment is fixed to be $c(p_i, l_j) = \text{distance}(p_i, l_j)$. A maximal matching with minimum cost is achieved by converting the problem to a flow network. In the flow network, the capacities of all the edges are set to one. Optimal solution is achieved by running Ford–Fulkerson's algorithm using minimum cost augmenting paths. The complexity of the algorithm is $O(p^2 \log p)$. Fig. 11 illustrates the flow network formed by the pin-to-location matching network.

IV. SOP Layer Assignment

A. Problem Formulation

For each routing interval i, the routing grid G_i contains all the (redistributed) pins from the top and bottom pin distribution layers ($L_t(i)$ and $L_b(i)$). We generate Steiner-tree-based routing topology to connect these pins during our topology generation step. The goal is to minimize the maximum sink delay $D^{\max}$ as discussed in Section II-B. The routing layer $L_r(i)$ in each routing interval i consists of several layer pairs, where each pair consists of one layer for horizontal wires and another layer for vertical wires. Thus, we can assign an entire rectilinear routing tree to a routing pair. In addition, two trees that are intersecting can also be assigned to the same routing pair provided that they do not violate the wire capacity of the routing regions involved. The SOP layer assignment problem is to assign each net to a routing

[6]Note that it is possible to redistribute all pins in each routing interval using network flow-based matching. However, due to a large number of pins and its prohibitive runtime, we adopt a hierarchical approach based on clustering.

```
SOP Layer Assignment
1:  for (each routing interval i)
2:      LCG_i = layer constraint graph for i;
3:      S = sort all nodes in LCG_i;
4:      color = 0;
5:      for (node j ∈ S)
6:          fin[j] = colors used by neighbors of j;
7:          tmp = lowest used color ∉ fin[j];
8:          if (tmp = ∅)
9:              color = color + 1;
10:             j ← color;
11:         else
12:             j ← tmp;
13:     assign a layer pair to each net;
```

Fig. 12. Pseudocode for SOP layer assignment algorithm.

layer pair so that the wire capacity constraint is satisfied and the total number of layer pairs used for all routing intervals is minimized.

For each routing interval i, we construct a Layer Constraint Graph (LCG) [43], denoted LCG_i, as follows: corresponding to each net in $n \in RI(i)$, we have a node in LCG_i. Two nodes $x, y \in \text{LCG}_i$ have an edge $e = (x, y)$ between them if net segments $s_x \in x$ and $s_y \in y$ are sharing the same edge in G_i, i.e., s_x and s_y are sharing the same boundary of a routing region. Then, we use a node coloring algorithm to assign colors to the nodes in LCG_i such that no two nodes sharing an edge are assigned the same color. Let C^{tot} denote the total number of colors used during node coloring, and let w denote the wire capacity of the boundary in routing region. Then, the total number of layer pairs used in this routing interval is computed as

$$|L_r(i)| = \left\lceil \frac{C^{tot}}{w} \right\rceil.$$

Lastly, a node with color $qw + r \ \ (r < w)$ is assigned to layer pair q. Let $h_{\max}$ and $v_{\max}$, respectively, denote the maximum number of wires used among all horizontal and vertical edges in G_i. Then, the following is a lower bound on the number of layers used in $L_r(i) : |L_r(i)| \geq \max\{h_{\max}, v_{\max}\}/w$.

We use the existing RSA/G heuristic [38] to optimize the performance of the routing topology. The minimum shortest path Steiner arborescence (MSPSA) generated by the heuristic guarantees the shortest path between every source-to-sink path and minimal overall weight of the tree. This routing topology is useful in high-performance SOP design due to its performance guarantee.

B. Layer Assignment Algorithm

Fig. 12 shows the SOP layer assignment algorithm that includes our coloring heuristic. We first sort all nodes in LCG_i in decreasing order of the number of their neighbors (line 3). Let $fin[n]$ denote the set of colors used by the neighbors of n (line 6). We visit the nodes in the sorted order (line 5). In case there exists a used color that is not included in $fin[n]$ (line 7), we assign this color to n (line 12). In case there exist multiple colors that satisfy this condition, we use the lowest color. Otherwise, we introduce a new color and assign it to n (line 9–10). Lastly, we assign a layer pair to each net based on its color (line 13). In spite of its simplicity, this greedy algorithm provides results that are very close to the lower bound on the total number of layers used as demonstrated in Section VI. The complexity of the SOP layer assignment algorithm is $O(N \log N)$, N is the total number of nets.

The generation of the layer conflict graph takes $O(N^2)$ time. For bigger circuits, the runtime and memory may be the issues that cannot be ignored. This issue is resolved by using net clustering. The nets are clustered into smaller sizes and layer assignment is performed on each of the clusters. The sum of the layer usage for each cluster gives a valid layer count. High connectivity between clusters ensures good quality. The heuristic used in this work sorts the nets based on their edge length and assigns them sequentially to the clusters. The effectiveness of the clustering heuristic can be measured by comparing the layer counts with the theoretical minimum. Clustering was used for two of the biggest benchmarks and the size of the clusters was fixed to four.

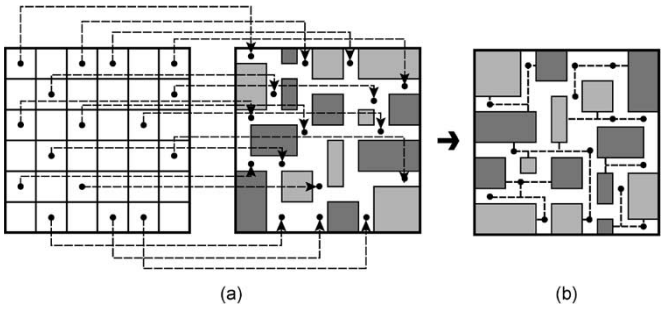

Fig. 13. Illustration of (a) channel assignment and (b) local routing.

V. SOP Channel Assignment

We first present an overview of our approach, followed by the problem formulation for SOP channel assignment and local routing. We use an existing method for local routing, so we focus on discussing our channel assignment heuristic in detail.

A. Overview of the Approach

Our prior topology generation and layer assignment steps focus on the connections among the distributed pins in the pin distribution layers using the routing layer pairs. After these steps are finished, there remain two kinds of connections: 1) connections between a pair of neighboring pin distribution and placement layers, i.e., connection between the original and the distributed pins and 2) connections between two nonneighboring pin distribution layers, i.e., feed-through via insertion. For both types of connections, the routing channels in the placement layers are used. Our strategy is to finish these remaining connections in two steps: channel assignment and local routing. During the channel assignment step, each pin in the pin distribution layer is mapped to a routing channel in the neighboring placement layer. In addition, each pin from an x-net that needs to penetrate a placement layer is also mapped to a routing channel in the placement layer.[7] Lastly, we generate routing topology for each pin-to-channel connection and assign it to a routing layer pair in the pin distribution layer. During the local routing, we finish connections between the pins now located in the routing channels and the pins on the block boundaries. An illustration is shown in Fig. 13.

B. Problem Formulation

For each placement layer $L_p(i)$, let $X(i)$ denote the set of pins that need to be mapped to a routing channel in $L_p(i)$. $X(i)$ contains pins from $L_b(i-1)$ and $L_t(i)$. The pins in $X(i)$ are grouped into two sets: terminal pin set $P_t(i)$ for the pins that have terminals in $L_p(i)$ and feed-through pin set $P_f(i)$ for the pairs of pins that require feed-through vias to penetrate $L_p(i)$. Let $C(i)$ denote the set of routing

[7]We perform channel assignment for the i-nets that connect to soft blocks only. The "pin assignment edges" in our FCG shown in Fig. 4 are used to determine which boundary each pin will be located. For the i-nets that connect to hard blocks that have pins along the boundaries, channel assignment is not necessary.

SOP Channel Assignment
1: **for** (each placement layer i)
2: $C(i)$ = channels in i;
3: $G(i)$ = 2D routing grid;
4: S = sorted feed-through pins;
5: **for** (each pin $p \in S$)
6: $best = \emptyset$;
7: **for** (each channel $c \in C(i)$);
8: compute $cost(p, c)$ and update $best$;
9: T_p = p-to-$best$ routing topology in $G(i)$;
10: update channel and edge usage in $G(i)$;
11: **repeat** line 4-10 for terminal pin set;
12: perform layer assignment on $G(i)$;

Fig. 14. Pseudocode for SOP channel assignment algorithm.

channels in $L_p(i)$. Each channel $c \in C(i)$ is associated with the pin capacity constraint. The goal of the SOP channel assignment problem is to map each pin $p \in X_i$ to a routing channel $c \in C(i)$ for $1 \leq i \leq K$ and finish p-to-c connection while satisfying the pin capacity constraint A, i.e., $|C(i)| < A$. For each pair of pins $(p_1, p_2) \in P_f(i)$, we map p_1 and p_2 to the same channel $c \in C(i)$. Let $|L_t(i)|$ denote the number of layers used to finish the connection between pins from $L_p(i)$ and $L_t(i)$, and let $|L_b(i)|$ denote the number of layers used to finish the connection between pins from $L_p(i+1)$ and $L_b(i)$.[8] The objective of SOP channel assignment is to minimize the cost function

$$w_1 \sum_{1 \leq i \leq K} (|L_t(i)| + |L_b(i)|) + \sum_{n \in NL} (w_2 wl_n + w_3 via_n).$$

Each placement layer is modeled with the FCG [36], as illustrated in Fig. 14. The input to the SOP local routing problem is a set of two-pin connections, where each connection is between a pin located in a routing channel and the other pin located along a block boundary. Each routing channel is associated with the wire capacity constraint, and pin assignment edge has a pin capacity for each boundary of soft block. The goal of SOP local routing is to 1) finish the routing for the given set of two-pin connections while satisfying the wire capacity constraint and 2) decide the location of pins for soft blocks along their boundaries. The objective is to minimize the total wire length, maximum pin demand, and maximum routing demand. Pin demand is the number of nets using the same block boundary, and routing demand is the number of nets using the same routing region. Both objectives have a direct relation to congestion in the 3-D structure of SOP.

We use the standard two-phase method for local routing: maze routing followed by congestion-driven rip-up-and-reroute [39]. During the first phase, a shortest weighted path in FCG is found for each connection while ignoring the actual channel usage, where weight is based on the combination of wire length and via. During the second phase, we rip-up nets that use the most congested channels and reroute it to alleviate the congestion problem.

C. SOP Channel Assignment

A pseudocode for the SOP channel assignment algorithm is shown in Fig. 14. We visit each placement layer and assign the feed-through pins and terminal pins to the channels. In addition, an L-shaped routing topology for each pin-to-channel is constructed in a 3-D grid $G(i)$ (line 3). The signal delay of feed-through vias is larger than that of other types of vias. Since each channel is under a capacity constraint, it is important to assign the feed-through pins to the nearest channels first. Therefore, our strategy is to perform channel assignment for the feed-through pins first to minimize the delay of x-nets that require feed-through vias. In addition, we give priority to the pins that are included in long nets. Thus, we sort the pins based on the wire length (line 4). Our heuristic algorithm assigns pins to channels based on the cost of mapping—we seek a channel with the best mapping cost for a given pin (lines 6–8). We compute the cost of mapping for a given pin p and a channel c as

$$\text{cost}(p, c) = \frac{\text{room}(c)}{\text{dist}(p, c) \times \text{bend}(p, c) \times \text{cong}(p, c)}$$

where $\text{room}(c)$ denotes the number of pins c it can accommodate until it violates the capacity constraint, $\text{dist}(p, c)$ is the Manhattan distance between p and c, $\text{bend}(p, c)$ is the number of bends in the connection between p and c, and $\text{cong}(p, c)$ denotes the total number of existing connections along the proposed L-shaped route. We choose the channel with the maximum $\text{cost}(p, c)$. Note that a channel is represented with a line instead of a point. Thus, $\text{distance}(p, c)$ and $\text{bend}(p, c)$ are based on the shortest connection between p and any point on c. Upon a pin to channel mapping, we update the usage of channel in $C(i)$ and edges in $G(i)$ (line 10). After the channel assignment and topology generation for all pin-to-channel connections are finished, we perform layer assignment using our coloring heuristic presented in Section IV-B and compute the total number of layers used. The complexity of the SOP channel assignment algorithm is $O(|P| \cdot |C|)$, where P and C denote the total number of pins and the channels in the given SOP design, respectively.

TABLE II
BENCHMARK CHARACTERISTICS. WE REPORT THE WIRE CAPACITY OF EACH ROUTING TILES (WCAP) AD THE 2-D GRID SIZE USED FOR OUR CPD AND DPD

ckts	blk	i-net	x-net	pin	wcap	CPD	DPD
n30	30	97	252	723	10	3 x 3	24 x 22
n50	50	76	409	1050	10	4 x 4	28 x 26
n100	100	189	696	1873	10	5 x 5	34 x 35
n200	200	297	1288	3599	10	6 x 6	46 x 48
n300	300	339	1554	4358	10	8 x 8	51 x 50
gt50	50	2102	7317	20835	20	4 x 4	110 x 101
gt100	100	4361	11823	36033	20	6 x 6	138 x 140
gt300	300	4534	15538	45901	20	8 x 8	163 x 165
gt1000	1000	6908	25561	79830	20	15 x 15	218 x 219
gt1500	1500	6554	28171	87785	20	15 x 15	225 x 227

VI. EXPERIMENTAL RESULTS

We implemented our algorithms in C++/STL and ran our experiments on Linux Beowulf clusters. We tested our algorithms with two sets of benchmarks. The first set is from standard GSRC floorplan circuits, where the hard blocks are placed into four-layer SOPs using our SOP floorplanner [4]. The second set, named the GT benchmark, was synthesized from IBM circuits [44], where we use our multilevel partitioner [42] to divide the gate-level netlist into multiple blocks first and then use our SOP floorplanner [4] to floorplan them to again four-layer SOPs. Table II shows the characteristics of the GSRC and GT benchmark designs. The GSRC benchmarks are small to medium sized in terms of both the number of blocks and the nets.[9] The GT benchmarks contain medium to large number of blocks with dense netlists. We note that in both cases the number of i-nets is only a small fraction of the total nets. The final area refers to the overall footprint area of the four-layer floorplan, which is determined by the maximum width and height among the individual floorplan layers.

[8]We assume that these layers are actually $x-y$ routing layer pairs that require via usage. In case of MCM pin redistribution, planar routing is done in these layers to avoid via congestion [18]–[20].

[9]The GT benchmark circuits are available for download at our website: http://www.gtcad.gatech.edu.

TABLE III
SOP PIN REDISTRIBUTION RESULTS. WE REPORT THE WIRE LENGTH BETWEEN THE ORIGINAL AND THE NEW LOCATION (dw), TOTAL WIRE LENGTH (wl), AND CROSSTALK (XTALK)

	DPD			CPD+DPD		
ckt	dw	wl	xtalk	dw	wl	xtalk
n30	0.15	0.07	0	0.16	0.07	0
n50	0.23	0.11	5	0.24	0.10	5
n100	0.41	0.18	10	0.41	0.16	10
n200	0.76	0.34	51	0.80	0.29	52
n300	1.13	0.50	219	1.18	0.44	220
gt50	1.96	0.87	92	1.98	0.71	90
gt100	4.10	1.78	441	4.08	1.44	446
gt300	5.71	2.48	2200	5.08	2.03	2186
gt1000	10.53	4.53	13365	10.75	3.72	13496
gt1500	12.55	5.58	14296	12.78	4.43	14366
TIME	529			547		

TABLE IV
TOPOLOGY GENERATION AND LAYER ASSIGNMENT RESULTS. WE REPORT THE TOTAL WIRE LENGTH (wl), ELMORE DELAY (dly), AND THE LOWER BOUND (LOW) AND THE ACTUAL NUMBER (LYR) OF LAYERS USED FOR THE TOPMOST ROUTING INTERVAL (EXCLUDING PIN REDISTRIBUTION LAYERS)

	RSA/G		LAYER	
ckt	wl	dly	low	lyr
n30	389	2.759	2	2
n50	404	2.872	3	3
n100	361	2.564	4	4
n200	378	2.687	6	6
n300	506	3.599	7	7
gt50	199	1.410	16	16
gt100	261	1.849	28	28
gt300	257	1.821	36	36
gt1000	331	2.346	26	30
gt1500	374	2.649	30	34
TIME	90		150	

In Table III, we compare pin redistribution results. In the DPD scheme, we skip CPD and perform net and DPD only. In CPD + DPD, we do not skip CPD. DPD serves as our baseline, where CPD + DPD demonstrates the impact of our CPD. The time reported is the average runtime among the GSRC/GT circuits. From the comparison between DPD and CPD + DPD, we note that the displacement result (dw) increases by an average of 1%. However, CPD lowers the total wire length (wl) consistently by 12% on average. The metric dw is the measure of routing from the redistributed pins to the originating pins.

In Table IV, we show our topology generation (RSA/G) and layer assignment results. We used the technology parameters for a 0.13-μm process for Elmore delay computation. Specifically, the driver resistance of 29.4 kΩ, input capacitance of 0.050 fF, unit length resistance of 0.82 Ω/μm, and unit length capacitance of 0.24 fF μm are used. We report the total wire length (wl), Elmore delay of the nets with maximum sink delay (dly), and the lower bound and the actual number of layers used for the topmost routing interval. In general, GSRC benchmarks have bigger delay than GT benchmarks due to the larger average wire length. Our layer assignment algorithm is able to achieve results very close to the lower bounds discussed in Section IV-A. For the GT circuits, the layer assignment results are within 10% of the lower bound. For the GSRC circuits, we were able to achieve results equal to the lower bound.

Our channel assignment results are shown in Table V for the baseline (wire length minimization only) and multiobjective algorithms. We observe that the number of layers is consistently and significantly reduced especially for the bigger GT benchmarks, where an average improvement of 38% is observed. In case of the second largest benchmark gt1000, we achieved 59% improvement. This saving on the layer usage comes at the cost of increase in wire length and vias. The average increase in wire length is 20% and 23% for GSRC and GT benchmarks, respectively. The average increase in via usage is 85%. The numbers of layers, wire length, and via are conflicting objectives. We noted that the channel assignment result is very sensitive to the weighting constants among the objectives used in our cost function. This indicates that the solution space of the channel assignment problem offers many useful tradeoff points. The primary objective for optimized channel assignment was the number of layers and the wire length. Via was the secondary objective.

TABLE V
SOP CHANNEL ASSIGNMENT RESULTS. WE REPORT THE LAYER USAGE (LYR), WIRE LENGTH (wl), AND VIA FOR ALL PIN DISTRIBUTION LAYERS

	wl-only			lyr+wl+via		
ckt	lyr	wl	via	lyr	wl	via
n30	8	0.030	107	8	0.033	144
n50	10	0.036	157	8	0.040	241
n100	9	0.038	303	8	0.047	470
n200	11	0.059	595	10	0.078	1231
n300	13	0.067	728	12	0.084	1292
gt50	11	0.383	3647	9	0.543	7772
gt100	16	0.893	7263	11	1.117	16133
gt300	25	1.291	12370	15	1.490	29285
gt1000	59	2.705	30869	24	3.220	68555
gt1500	63	3.430	38869	35	3.971	81093
TIME	194			210		

TABLE VI
SOP LOCAL ROUTING RESULTS. WE REPORT THE WIRE LENGTH (wl), MAXIMUM (MAX), AND AVERAGE (AVE) ROUTING DEMAND AS WELL AS THE STANDARD DEVIATION (DEV)

	wl-only				wl+rd			
ckt	wl	max	avg	dev	wl	max	avg	dev
n30	0.08	50	7.8	9.2	0.08	39	7.5	7.5
n50	0.13	68	10.1	11.3	0.13	61	9.9	9.5
n100	0.25	206	15.2	20.1	0.28	117	15.2	15.0
n200	0.51	316	22.4	29.1	0.59	146	22.8	19.6
n300	0.78	318	22.4	29.4	0.92	146	23.2	19.4
gt50	1.41	3428	296.7	455.7	1.68	2100	311.0	358.5
gt100	2.78	4479	341.8	604.5	3.31	3308	364.7	459.7
gt300	5.12	6702	331.1	609.4	6.27	3721	362.9	442.7
gt1000	8.14	6460	264.7	500.6	9.68	3757	305.1	383.3
gt1500	9.59	9221	256.5	504.7	11.80	4259	292.9	377.1
TIME	547				573			

Table VI reports our SOP local routing results for the baseline (wire length minimization only) and multiobjective algorithms. In both cases, the same pin demand constraint is imposed. We note that the improvement of our multiobjective algorithm over the baseline is significant, especially for GSRC circuits—the routing demands were reduced by 35% on average while the wire length increased by only 14%. In addition, we reduced the routing demands for the GT benchmarks by 37% on average, with wire length increase by 10%. In our biggest benchmarks (gt1500), our routing demand reduction is the largest (54%), which comes with the maximum increase in wire length (23%). This again indicates that the local routing result is very sensitive to the weighting constants among the objectives used in our cost function. The lower standard deviation of our multiobjective algorithm indicates that the routing demand is more evenly distributed (= lower congestion) compared to the wire-length-only case.

VII. CONCLUSION AND ONGOING WORKS

In this paper, we presented 3PGR, the first global routing algorithm for 3-D packaging via SOP. We formulated the new pin redistribution, net distribution, and channel assignment problems that are unique in

3-D packaging designs compared with the traditional MCM designs. We provided detailed discussions on how SOP routing is different from MCM routing and provided various routing resource models. We are currently looking at thermal-aware global routing. Our 3-D global router is currently being integrated into our 3-D microarchitecture design space exploration framework [45] for more accurate performance and power measurements. Lastly, we are working on detailed routing for 3-D packaging.

REFERENCES

[1] R. Tummala and V. Madisetti, "System on chip or system on package?" *IEEE Des. Test Comput.*, vol. 16, no. 2, pp. 48–56, Apr. 1999.

[2] R. Tummala, "SOP: What is it and why? A new microsystem-integration technology paradigm—Moore's law for system integration of miniaturized convergent systems of the next decade," *IEEE Trans. Adv. Packag.*, vol. 27, no. 2, pp. 241–249, May 2004.

[3] S. K. Lim, "Physical design for 3D system-on-package: Challenges and opportunities," *IEEE Des. Test. Comput.*, vol. 22, no. 6, pp. 532–539, 2005.

[4] P. Shiu, R. Ravichandran, S. Easwar, and S. K. Lim, "Multi-layer floorplanning for reliable system-on-package," in *Proc. IEEE Int. Symp. Circuits and Systems*, Vancouver, BC, Canada, 2004, pp. V-69–V-72.

[5] R. Ravichandran, J. Minz, M. Pathak, S. Easwar, and S. K. Lim, "Physical layout automation for system-on-packages," in *IEEE Electronic Components and Technology Conf.*, Las Vegas, NV, 2004, pp. 41–48.

[6] J. Minz, S. K. Lim, J. Choi, and M. Swaminathan, "Module placement for power supply noise and wire congestion avoidance in 3D packaging," in *Proc. IEEE Electrical Performance Electronic Packaging*, Portland, OR, 2004, pp. 123–126.

[7] J. Minz and S. K. Lim, "A global router for system-on-package targeting layer and crosstalk minimization," in *Proc. IEEE Electrical Performance Electronic Packaging*, Portland, OR, 2004, pp. 99–102.

[8] ——, "Layer assignment for system on packages," in *Proc. Asia and South Pacific Design Automation Conf.*, Yokohama, Japan, 2004, pp. 31–37.

[9] J. Minz, M. Pathak, and S. K. Lim, "Net and pin distribution for 3D package global routing," in *Proc. Design, Automation and Test Europe*, Paris, France, 2004, pp. 1410–1411.

[10] J. Minz, E. Wong, and S. K. Lim, "Thermal and congestion-aware physical design for 3D system-on-package," in *IEEE Electronic Components and Technology Conf.*, Orlando, FL, 2005, pp. 824–831.

[11] J. Minz, E. Wong, M. Pathak, and S. K. Lim, "Placement and routing for 3D system-on-package designs," *IEEE Trans. Compon. Packag. Technol.*, 2006, to be published.

[12] K. Khoo and J. Cong"An efficient multilayer MCM router based on four-via routing," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 14, no. 10, pp. 1277–1290, Oct. 1995.

[13] J. D. Cho, K. Liao, S. Raje, and M. Sarrafzadeh, "$\mathrm{M}^2\mathrm{R}$: Multilayer routing algorithm for high-performance MCM," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 41, no. 4, pp. 253–265, Apr. 1994.

[14] W. W. Dai, "Topological routing in surf: Generating a rubberband sketch," in *Proc. ACM Design Automation Conf.*, San Francisco, CA, 1991, pp. 39–48.

[15] D. Wang and E. Kuh, "A new timing-driven multilayer MCM/IC routing algorithm," in *Proc. IEEE Multi-Chip Module Conf.*, Santa Cruz, CA, 1997, pp. 89–94.

[16] Y. Cha, C. Rim, and K. Nakajima, "A simple and effective greedy multilayer router for MCMs," in *Proc. Int. Symp. Physical Design*, Napa Valley, CA, 1997, pp. 67–72.

[17] J. Cong and P. Madden, "Performance driven multi-layer general area routing for PCB/MCM designs," in *Proc. ACM Design Automation Conf.*, San Francisco, CA, 1998, pp. 356–361.

[18] J. Cho and M. Sarrafzadeh, "The pin redistribution problem in multi-chip modules," in *Proc. Int. ASIC Conf.*, Rochester, NY, 1991, pp. 9-2-1–9-2-4.

[19] D. Chang, T. Gonzales, and O. Ibarra, "A flow based approach to the pin redistribution problem for multi-chip modules," in *Proc. Great Lakes Symp. VLSI*, Notre Dame, IN, 1994, pp. 114–119.

[20] J. D. Cho, "An optimum pin redistribution for multichip modules," in *Proc. IEEE Multi-Chip Module Conf.*, Santa Cruz, CA, 1996, pp. 111–116.

[21] S. Das, A. Chandrakasan, and R. Reif, "Design tools for 3-D integrated circuits," in *Proc. Asia and South Pacific Design Automation Conf.*, Kitakyushu, Japan, 2003, pp. 53–56.

[22] B. Goplen and S. Sapatnekar, "Efficient thermal placement of standard cells in 3D ICs using a force directed approach," in *Proc. IEEE Int. Conf. Computer-Aided Design*, San Jose, CA, 2003, pp. 86–90.

[23] T. Tanprasert, "An analytical 3-D placement that reserves routing space," in *Proc. IEEE Int. Symp. Circuits and Systems*, Geneva, Switzerland, 2000, pp. 69–72.

[24] R. Zhang, K. Roy, C.-K. Koh, and D. B. Janes, "Exploring SOI device structures and interconnect architectures for 3-dimensional integration," in *Proc. ACM Design Automation Conf.*, Las Vegas, NV, 2001, pp. 846–851.

[25] K. Balakrishnan, V. Nanda, S. Easwar, and S. K. Lim, "Wire congestion and thermal aware 3D global placement," in *Proc. Asia and South Pacific Design Automation Conf.*, Shanghai, China, 2005, pp. 1131–1134.

[26] J. Minz, E. Wong, and S. K. Lim, "Thermal and power integrity-aware floorplanning for 3D circuits," in *Proc. IEEE Int. SOC Conf.*, Washington, DC, 2005, pp. 81–82.

[27] L. Cheng, W. Hung, G. Yang, and X. Song, "Congestion estimation for 3-D circuit architectures," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 51, no. 12, pp. 655–659, Dec. 2004.

[28] B. Goplen and S. Sapatnekar, "Thermal via placement in 3-D ICs," in *Proc. Int. Symp. Physical Design*, San Francisco, CA, 2005, pp. 167–174.

[29] V. Pavlidis and E. Friedman, "Interconnect delay minimization through interlayer via placement in 3-D ICs," in *Proc. Great Lakes Symp. VLSI*, Chicago, IL, 2005, pp. 20–25.

[30] I. Kaya, M. Olbrich, and E. Barke, "3-D placement considering vertical interconnects," in *Proc. IEEE Int. SOC Conf.*, Tampere, Finland, 2003, pp. 257–258.

[31] Y. Deng and W. Maly, "Physical design of the 2.5D stacked system," in *Proc. IEEE Int. Conf. Computer Design*, San Jose, CA, 2003, pp. 211–217.

[32] J. Cong, J. Wei, and Y. Zhang, "A thermal-driven floorplanning algorithm for 3D ICs," in *Proc. IEEE Int. Conf. Computer-Aided Design*, San Jose, CA, 2004, pp. 306–313.

[33] J. Cong and Y. Zhang, "Thermal-driven multilevel routing for 3-D ICs," in *Proc. Asia and South Pacific Design Automation Conf.*, Shanghai, China, 2005, pp. 121–126.

[34] L. Cheng, L. Deng, and M. Wong, "Floorplan design for 3-D VLSI design," in *Proc. Asia and South Pacific Design Automation Conf.*, Shanghai, China, 2005, pp. 405–411.

[35] J. Minz, S. K. Lim, and C. K. Koh, "3D module placement for congestion and power noise reduction," in *Proc. Great Lakes Symp. VLSI*, Chicago, IL, 2005, pp. 458–461.

[36] J. Cong, "Pin assignment with global routing for general cell design," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 10, no. 11, pp. 1401–1412, Nov. 1991.

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

[38] J. Cong, A. B. Kahng, and K.-S. Leung, "Efficient algorithms for the minimum shortest path Steiner arborescence problem with applications to VLSI physical design," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 17, no. 1, pp. 24–39, Jan. 1998.

[39] L. McMurchie and C. Ebeling, "Pathfinder: A negotiation based performance-driven router for FPGAs," in *ACM Int. Symp. Field-Programmable Gate Arrays*, Monterey, CA, 1995, pp. 111–117.

[40] J. D. Cho, S. Raje, M. Sarrafzadeh, M. Sriram, and S. M. Kang, "Crosstalk-minimum layer assignment," in *IEEE Custom Integrated Circuits Conf.*, San Diego, CA, 1993, pp. 29.7.1–29.7.4.

[41] J. Cong and S. K. Lim, "Retiming-based timing analysis with an application to mincut-based global placement," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 23, no. 12, pp. 1684–1692, Dec. 2004.

[42] ——, "Edge separability based circuit clustering with application to multi-level circuit partitioning," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 23, no. 3, pp. 346–357, Mar. 2004.

[43] M. Ho, M. Sarrafzadeh, G. Vijayan, and C. Wong, "Layer assignment for multichip modules," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 9, no. 12, pp. 1272–1277, Dec. 1990.

[44] C. J. Alpert, "The ISPD98 circuit benchmark suite," in *Proc. Int. Symp. Physical Design*, Monterey, CA, 1998, pp. 80–85.

[45] M. Ekpanyapong, J. Minz, T. Watewai, H.-H. Lee, and S. K. Lim, "Profile-guided microarchitectural floorplanning for deep submicron processor design," in *Proc. ACM Design Automation Conf.*, San Diego, CA, 2004, pp. 634–639.