Routing Complexity Minimization of Monolithic Three-Dimensional Integrated Circuits

Sheng-En (David) Lin and Dae Hyun Kim
School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA, USA
Email: {slin3, daehyun}@eecs.wsu.edu

***Abstract*—Monolithic three-dimensional (3D) integration provides the most fine-grained integration of transistors. Monolithic inter-tier vias (MIVs) used for inter-tier electrical connections in monolithic 3D integrated circuits (ICs) are as small as local vias, so parasitic resistance and capacitance of an MIV is much smaller than that of a through-silicon via (TSV). In addition, MIVs are much smaller than TSVs, so many MIVs can be inserted into a layout while enabling very high bandwidth between adjacent tiers. Thus, monolithic 3D ICs have been actively researched in the literature recently. In this paper, we proposes a gate composition algorithm to reduce the runtime for routing of monolithic 3D ICs. Simulation results show that the gate composition algorithm reduces the runtime by 11% to 76%.**

I. Introduction

Three-dimensional (3D) integration provides many benefits such as wirelength reduction, higher performance, smaller footprint area, and much higher inter-tier (or inter-die)[1] bandwidth than two-dimensional integrated circuits (2D ICs). To build 3D ICs, various 3D integration technologies have been proposed and developed in academia and industry. Two representative 3D integration technologies are silicon-interposer-based [19] and through-silicon-via (TSV)-based 3D ICs [6], [7]. Especially, TSV-based 3D integration stacks all dies in a single package and internally connects them by TSVs, so it provides more benefits than the silicon-interposer-based 3D integration.

An issue of the TSV-based 3D integration is that TSVs occupy non-negligible silicon area. A typical diameter of the state-of-the-art TSVs is still around 1um, which is comparable to the height of a logic gate in a 45nm technology. Therefore, inserting a proper amount of TSVs becomes very important in the design of TSV-based 3D ICs to reduce the wirelength and improve the performance without increasing the die area. According to [5], however, inserting too many TSVs into a layout increases its wirelength significantly because of serious area overhead and inserting too few TSVs does not sufficiently reduce the wirelength. Therefore, design of TSV-based 3D ICs requires TSV-aware placement algorithms, which are still under research [4], [10].

Monolithic 3D integration provides more dense 3D integration than the TSV-based 3D integration because monolithic inter-tier vias (MIVs) are much smaller than TSVs [1]. Thus, it is expected that inserting as many MIVs as we want is allowed in the design of monolithic 3D ICs, which is quite different from the design of TSV-based 3D ICs for which the TSV count should be controlled carefully. Therefore, monolithic 3D integration is expected to enable the highest degree of wirelength and footprint area reduction, performance improvement, and power reduction.

Routing of monolithic 3D IC layouts (3D routing) can route all the nets of a given design simultaneously or sequentially. For example, Panth uses a simultaneous 3D routing methodology based on library modification [13]. On the other hand, most of the other papers in the literature use sequential 3D routing methodologies, which first insert MIVs, then route 2D nets [8], [14]. Since the former routes 2D and 3D nets simultaneously using a commercial tool, it could optimize the wirelength more effectively than the latter. However, the runtime of the simultaneous 3D routing methodology increases significantly as the number of tiers goes up because the number of routing layers also increases and the router should handle more routing blockages.

In this paper, we apply a gate composition algorithm to reduce the runtime of the simultaneous 3D routing methodology for the design of three- and four-tier monolithic 3D IC layouts.

II. Preliminaries

In this section, we review previous work on the design of monolithic 3D ICs and the impact of the use of 2D and 3D standard cells on the quality of monolithic 3D ICs.

A. Previous Work

Liu presented two-tier monolithic 3D IC designs in [2] in which the authors compared 2D- and 3D-cell-based monolithic 3D ICs. The 2D-cell-based design, so called gate-level monolithic integration (G-MI), places 2D cells in two tiers. On the other hand, the 3D-cell-based design, so called transistor-level monolithic integration (T-MI), places 3D cells in which transistors are placed in two tiers and connected by MIVs. Since 3D cells can be treated as 2D cells for 3D placement in the two-tier 3D IC design, the authors used a commercial placement tool to build T-MI designs. However, the authors used an in-house placement tool to build G-MI designs because the commercial placement tool handled only a single tier, so they could not directly compare T-MI and G-MI designs. Bobba also introduced 3D cells for monolithic 3D IC design in [15].

[1] In this paper, we use "dies" and "tiers" for the silicon layers in non-monolithic and monolithic 3D ICs, respectively.

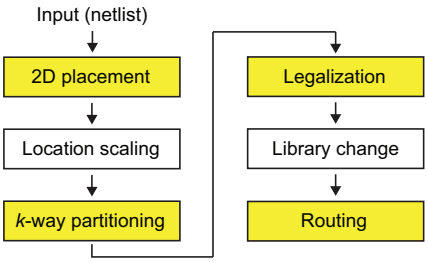

Fig. 1. Our design methodology for multitier gate-level monolithic 3D ICs. Commercial tools are used in the colored steps.

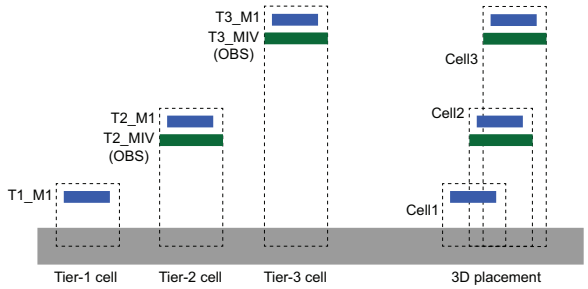

Fig. 2. 3D standard cells and the representation of 3D placement with the 3D standard cell library.

Lee extended the two-tier monolithic 3D IC design work in [20] in which the authors designed and optimized monolithic 3D IC layouts and obtained timing and power values. However, this work also has the same limitation as [2], which is that they used a commercial and in-house placement tools to build T-MI and G-MI designs, respectively. Panth proposed a new design methodology to design two-tier low-power G-MI 3D ICs in [17], [18]. In the work, the authors used a commercial placement tool to place 2D standard cells.

B. 2D vs. 3D Standard Cells

A few papers introduced 3D standard cells for T-MI designs [2], [15], [16], [20]. 2D standard cells use poly and metal1 layers in a tier, whereas 3D standard cells use poly and metal1 layers in both the bottom and the top tiers, and use via stacks for inter-tier intra-cell routing. By placing PFETs and NFETs in different tiers, the footprint area of a 3D standard cell is reduced by almost 35% to 40%.

However, 3D standard cells have two problems. First, the total chip area (# tiers $\times$ footprint area) of a design using 3D standard cells is greater than that using 2D standard cells because the footprint area of each 3D standard cell is greater than a half of the area of its 2D counterpart. Since PFETs are usually drawn larger than NFETs due to the mobility mismatch, the PFET area is larger than the NFET area. However, they are vertically aligned in the 3D standard cell boundary, so the total chip area of a monolithic 3D design using 3D standard cells is greater than that using 2D standard cells. More importantly, via stacks used for inter-tier intra-cell routing are routing blockages, so having too many via stacks in each 3D standard cell causes serious routing congestion. This problem has been reported in a few papers [2], [20]. To resolve the routing congestion problem, the authors of [20] tried reducing the width of the metal layers, but it did not completely fix design rule violations during routing. In this paper, therefore, we use 2D standard cells, which does not require re-designing 3D standard cells and cause the routing-blockage issues.

III. Design of Gate-Level Monolithic 3D ICs

In this section, we review the design methodologies proposed for the design of gate-level monolithic 3D ICs in [13] and [9]. Figure 1 shows the overall design methodology in [9].

A. 2D Placement and Location Scaling

The input to the placement methodologies is a synthesized netlist and standard cell libraries. With these files, the methodologies place the cells in the netlist in a 2D layout. To place the cells, the placement methodology proposed in [13] reduces the widths and the heights of the cells in the libraries by $1/\sqrt{T}$ where T is the number of tiers, and places the cells in a 2D layout using a commercial software. Once the cells are placed, the design is reloaded into the software with the original standard cell libraries. The placement methodology proposed in [9] places the cells in a 2D layout using a commercial software. Once the cells are placed, the coordinates of the cells and the width and height of the 2D layout are reduced by $1/\sqrt{T}$. The 2D layout obtained from the two methodologies have two problems. First, the cells overlap with each other. Second, the locations of the cells are not legal, i.e., they are not aligned with standard cell rows and columns. Thus, the cell locations are legalized by z-directional partitioning and cell snapping.

B. Partitioning

As mentioned above, the overlap problem is resolved by k-way partitioning, which is actually placing the cells in multiple tiers. k is set to T and the cells placed in partition p ($1 \leq p \leq T$) belong to Tier p. An issue in the partitioning is that partitioning the whole netlist can result in a very unbalanced placement result. Since the locations of the cells are optimized by the commercial software, the partitioning should remove the cell overlaps only locally and should not perturb the 2D placement result significantly, otherwise the wirelength will not be optimized well. Thus, the partitioning step should balance the cell area of the tiers globally and locally at the same time. For this, the partitioning step splits the layout into an $n \times n$ grid and runs partitioning in each grid considering the partitions (tiers) of the cells already partitioned in other grids for globally-balanced partitioning.

C. From Legalization to Routing

Once the partitioning is completed, the cell locations are legalized by a commercial software in each tier separately. After legalization, the layout is reloaded into a commercial software with modified libraries [13]. The modified libraries have $M \cdot T$ metal layers where M is the number of metal

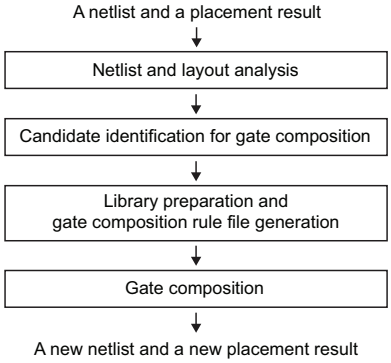

Fig. 3. Our design flow for the gate composition for routing complexity minimization.

layers in a tier. For instance, the metal 1 layer in Tier 1 is named "T1_M1" and that in Tier 2 is named "T2_M1" as shown in Figure 2. In addition, a cell has T definitions, one for each tier, in the modified libraries. For example, the smallest two-input NAND gate, NAND2_X1, has three definitions, NAND2_X1_T1, NAND2_X1_T2, NAND2_X1_T3, when there are three tiers ($T = 3$). The definition for NAND2_X1_Tn uses the Tn_M1 layer for the metal 1 layer of the cell. The given original netlist is also modified based on the partitioning result. For example, if cell U1 of type NAND2_X1 is partitioned to Tier 2, the type of the cell in the netlist is changed from NAND2_X1 to NAND2_X1_T2. Then, the reloaded design has all the cells in a 2D layout and the coordinates of the cells overlap, but the internal wires of the cells do not overlap. Thus, routing the design using a commercial software connects all the pins in different tiers.[2]

An issue in the 3D routing is that routers might insert vias (MIVs) inside cells placed in all the tiers except Tier 1, which should be forbidden. Thus, the definition of each cell in the modified libraries also has a blockage (obstruction) of the same size as the cell in the via tier defined right below the metal 1 tier of the tier the cell is placed in. In Figure 2, for example, the Tier-2 and Tier-3 cell definitions include obstructions in the T2_MIV and T3_MIV layers, respectively. *Due to these routing blockages, however, the runtime of 3D routing increases significantly as the tier count goes up.* For example, routing of a 2D 1,024-bit Kogge-Stone adder takes 157 seconds, but routing of its four-tier version takes 17,543 seconds, which is 112× slower, in our simulation.

IV. Routing Complexity Minimization

In this section, we present a gate-composition algorithm to reduce routing complexity.

A. Why Gate Composition?

Among various algorithms that could reduce the routing complexity of gate-level monolithic 3D IC layouts, we propose to use gate composition for the following reasons. First, the placement methodologies in [9], [13] reduce the wirelength

[2]Some commercial routers support routing of overlapped cells.

Algorithm 1: the proposed gate composition algorithm.

Input: Netlist N, 2D layout L, and gate-composition rules R.
Output: A new netlist and a new 2D layout.

```
1  for each cell c1 ∈ N do
2      if fanout(c1) = 1 then
3          c2 ← out_cell(c1) ;
4          if ∃r ∈ R where r consists of type(c1) and type(c2)
           then
5              C_in ← c ∈ in_cell(c2), c ≠ c1 ;
6              if Check_Location(c1, c2, C_in) = ok then
7                  Gate_Comp (r, c1, c2, C_in) ;
8              end
9          end
10     end
11 end
```

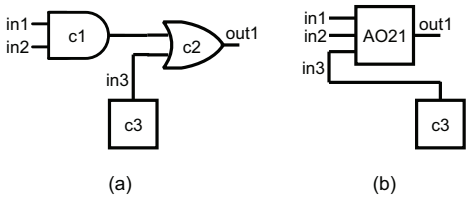

Fig. 4. Examples of the gate composition. "AO" denotes AND-OR.

effectively. If a net is long, decomposing a cell in the net into multiple sub-cells (e.g., decomposing an AND gate into a NAND gate and an inverter) and spreading them along the net reduces the net delay. However, since nets become shorter in monolithic 3D ICs, composing multiple sub-cells into a super-cell could reduce the routing complexity of a given design without timing degradation by reducing the cell and net counts.

B. Gate Composition Algorithm

Figure 3 shows the overall design flow for the gate composition for routing complexity reduction. For a given netlist and a 2D placement result, we analyze them and find potential candidates for gate composition. If some composite cells we need for the given netlist and layout are missing in a given standard cell library, we draw layouts for the composite cells, run RC extraction, and create a library for them and add the library to the given library (this step is represented as "Library preparation" in Figure 3). After we prepare standard cell libraries for the missing composite cells, we run gate composition for each candidate cell. Notice that some potential candidates selected for gate composition in this library preparation step may not be merged during the actual gate composition step as explained in the following paragraph.

Algorithm 1 shows our gate composition algorithm. For each cell in a given netlist, we select only fanout-1 cells, i.e., the output of the cell is connected to only one cell. This can be extended to more general cases where the output of a cell is connected to multiple cells, but we used simplified gate composition rules in this paper. For each fanout-1 cell, c_1, we find cell c_2 connected to the output of c_1 (out_cell(c_1) in the algorithm). For the cell types of c_1 and c_2, we search

TABLE I
METAL LAYERS USED FOR THE DESIGN OF 2D AND MONOLITHIC 3D ICS IN THIS PAPER. THE NUMBERS IN THE PARENTHESES AFTER "3D" DENOTE THE NUMBER OF METAL LAYERS USED IN EACH TIER. THE UNIT OF THE WIDTH AND PITCH IS NM.

2D			Two-tier 3D (6/6)			Three-tier 3D (4/4/4)			Four-tier 3D (4/4/4/3)		
Layer	Width	Pitch	Layer	Width	Pitch	Layer	Width	Pitch	Layer	Width	Pitch
M1 - M4	70	140	T1_M1 - M3	70	140	T1_M1 - M3	70	140	T1_M1 - M3	70	140
M5 - M6	140	280	T1_M4 - M6	140	280	T1_M4	140	280	T1_M4	140	280
M7 - M8	400	800	T2_M1 - M3	70	140	T2_M1 - M3	70	140	T2_M1 - M3	70	140
M9 - M10	800	1600	T2_M4 - M6	140	280	T2_M4	140	280	T2_M4	140	280
						T3_M1 - M3	70	140	T3_M1 - M3	70	140
						T3_M4	140	280	T3_M4	140	280
									T4_M1 - M3	70	140

TABLE II
BENCHMARKS USED IN THIS PAPER.

Circuit	Characteristics
Brent-Kung Adder	Max. logic depth, min. area
Han-Carlson Adder	Hybrid of Brent-Kung and Kogge-Stone Adders
Kogge-Stone Adder	Fast, min. fan-out, large area, routing congestion
Ladner-Fischer Adder	Min. logic depth, high fan-out

a given gate-composition rule deck R. If there exists a gate-composition rule for the cell types, we obtain the set C_{in} of all the cells connected to the inputs of c_2. We exclude c_1 from C_{in} because we will use C_{in} to check the locations of the other cells connected to c_2. Figure 4 shows an example. In the figure, c_3 is the only other cell connected to the inputs of c_2. If c_3 is located near c_1, the gate composition shown in Figure 4(b) does not lead to wirelength overhead. If c_3 is located far away from c_1 and around net out1, however, net in3 connecting c_3 and the new cell (AO21) will result in serious wirelength overhead. To avoid this wirelength overhead, we check the locations of c_1, c_2, and all the cells in C_{in} in function $Check\_Location()$ in Algorithm 1. $Check\_Location()$ first finds the bounding box containing all the cells in C_{in} and computes the difference between the distances from the bounding box to c_2 and c_1. If the difference is less than a pre-determined value (e.g., 20um), we apply gate composition to c_1 and c_2. Otherwise, we drop this candidate. If a cell can be merged into two different cells, we randomly select one of the two choices. For example, if c_3 in Figure 4(a) is also a two-input AND gate, we can merge (c_1 and c_2) or (c_3 and c_2).

V. SIMULATION RESULTS

We used Synopsys Design Compiler for netlist synthesis, Cadence Encounter for 2D placement, legalization, and routing, and hMetis for *k*-way partitioning. We used Nangate 45nm Open Cell Library [11] with NCSU 45nm FreePDK [12] for layout generation. Table I shows the metal layers used for the design of 2D and monolithic 3D ICs. Since no paper about monolithic 3D integration technologies showed details on their metal layers, we used metal layers similar to [20] for two-tier designs. To use the same (or similar) total number of metal layers, we used 10, 12, 12, and 15 metal layers for 2D, two-tier, three-tier, and four-tier designs, respectively.

A. Benchmarks

We use four different types of adders (Brent-Kung, Han-Carlson, Kogge-Stone, and Ladner-Fischer) with five bit widths (64, 128, 256, 512, and 1024). We use the adders for the benchmarks because the adders are widely used in numerous applications. In addition, they have a wide range of routing complexity, so we can also study the impact of routing complexity on the quality of 2D and 3D ICs. Table II shows the four adder types and brief description on the characteristics of the architectures of the adders.

B. Comparison of Footprint Area

We first compare the footprint area of 2D and monolithic 3D ICs. Table III shows the footprint area of all the 2D designs and the ratio of the footprint area between the 3D and the 2D designs. When we build an n-tier design, we set the core area of the n-tier design to A_{2D}/n where A_{2D} is the core area of its 2D design, so ideally the ratio of the footprint area between the 2D and the two-, three-, and four-tier designs should be 0.5, 0.333, and 0.25, respectively. However, we round up the height of the core area to align it with an integer multiple of the standard cell height, so the footprint areas of all the 3D designs are slightly larger than the ideal values. In general, however, the footprint area ratio approaches the ideal value as the circuit size goes up.

Two exceptional cases in Table III are the four-tier designs of the 1024-bit Kogge-Stone and Ladner-Fischer adders. They required a slightly larger scaling ratio than $1/\sqrt{n}$ because of high routing complexity. If they are designed in more than four tiers, routing complexity will go up and we will have to increase the area further. Thus, even with monolithic 3D integration technology, achieving the ideal footprint area ratio is sometimes not possible due to routing complexity.

C. Comparison of Wirelength

The wirelength of an n-tier 3D design is expected to be ideally $L_{\text{2D}}/\sqrt{n}$ where L_{2D} is the wirelength of its 2D counterpart [3]. When the design is sufficiently large (512-bit and 1024-bit), the wirelengths of the two-tier designs approach the ideal values in the case of the Brent-Kung, Han-Carlson, and Kogge-Stone adders as shown in Table III. However, the wirelength ratios of the 512-bit and 1024-bit Ladner-Fischer adders do not approach the ideal values due to routing congestion.

TABLE III
FOOTPRINT AREA (FP) AND WIRELENGTH (WL) RATIOS OF THE 2D, TWO-TIER, THREE-TIER, AND FOUR-TIER MONOLITHIC 3D DESIGNS.

Circuit	# Bits	# Nets	2D		Two-tier 3D		Three-tier 3D		Four-tier 3D	
			FP (um^2)	WL (um)	FP	WL	FP	WL	FP	WL
Brent-Kung	64	621	1,296 (1.000)	2,589 (1.000)	0.610	0.926	0.391	0.922	0.316	0.959
	128	1,258	2,500 (1.000)	6,653 (1.000)	0.546	0.876	0.371	0.853	0.295	0.847
	256	2,535	4,761 (1.000)	14,937 (1.000)	0.545	0.874	0.360	0.832	0.242	0.797
	512	5,092	9,216 (1.000)	43,294 (1.000)	0.530	0.852	0.357	0.771	0.272	0.719
	1,024	10,209	17,956 (1.000)	133,586 (1.000)	0.523	0.777	0.351	0.671	0.266	0.616
Han-Carlson	64	837	1,764 (1.000)	4,036 (1.000)	0.582	0.888	0.399	0.840	0.305	0.867
	128	1,861	3,721 (1.000)	11,162 (1.000)	0.543	0.843	0.377	0.783	0.277	0.762
	256	4,101	7,744 (1.000)	30,442 (1.000)	0.527	0.809	0.354	0.727	0.274	0.697
	512	8,965	16,641 (1.000)	86,728 (1.000)	0.518	0.783	0.351	0.695	0.262	0.635
	1,024	19,461	35,721 (1.000)	248,790 (1.000)	0.517	0.771	0.347	0.665	0.258	0.593
Kogge-Stone	64	1,267	2,601 (1.000)	8,122 (1.000)	0.555	0.851	0.381	0.788	0.283	0.764
	128	2,901	5,776 (1.000)	23,847 (1.000)	0.542	0.803	0.374	0.733	0.279	0.660
	256	6,551	12,769 (1.000)	69,560 (1.000)	0.525	0.788	0.356	0.685	0.264	0.623
	512	14,617	28,224 (1.000)	208,820 (1.000)	0.518	0.769	0.350	0.656	0.262	0.583
	1,024	32,283	62,500 (1.000)	663,671 (1.000)	0.512	0.735	0.343	0.605	0.310	0.584
Ladner-Fischer	64	837	1,764 (1.000)	3,639 (1.000)	0.582	0.975	0.399	0.970	0.305	0.949
	128	1,861	3,721 (1.000)	9,280 (1.000)	0.543	0.892	0.377	0.885	0.277	0.849
	256	4,101	7,921 (1.000)	23,640 (1.000)	0.532	0.890	0.360	0.850	0.268	0.832
	512	8,965	16,384 (1.000)	62,575 (1.000)	0.527	0.876	0.356	0.844	0.266	0.809
	1,024	19,461	35,344 (1.000)	167,409 (1.000)	0.515	0.857	0.345	0.808	0.295	0.772

Two exceptional cases are the four-tier 1024-bit Kogge-Stone and Ladner-Fischer adders. These circuits have serious routing congestion, so the footprint areas of these two designs are slightly larger than the ideal values. Due to the serious routing congestion and the increased footprint area, the wirelength ratios of these two designs are close to those of their three-tier counterparts.

D. Comparison of Runtime and # MIVs

Table IV shows runtimes for routing of the 2D and 3D designs. The first noticeable result is that the runtime for routing increases rapidly as the tier count goes up. For example, the runtime for routing of the 1024-bit adders designed in two tiers is $12\times$ to $22\times$ as large as the runtime for routing of their 2D counterparts. However, the runtime for routing of the adders designed in three tiers is $78\times$ to $110\times$ as large as that for routing of 2D designs. Since we are using the same number of metal layers (12 layers) for both the two- and three-tier designs, we find that designing monolithic 3D ICs in more tiers makes the routing more difficult. In addition, the runtime for routing of the 1024-bit adders designed in four tiers is $112\times$ to $255\times$ as large as that for routing of their 2D counterparts. We find from this comparison that the routing time increases super-linearly as we increase the tier count. Table IV also shows the number of MIVs used in each tier. As the table shows, the number of MIVs is roughly equally-distributed across the tiers.

E. Routing Complexity Reduction by Gate Composition

Table V compares the wirelength and runtime for routing without/with gate composition only for 512-bit and 1,024-bit designs due to page limit. As the table shows, the wirelengths of the designs optimized by gate composition are similar to those of the non-optimized designs. However, the runtime for routing of the designs optimized by the gate composition algorithm decreases significantly. For the most complex design (1,024-bit Kogge-Stone adder), the routing time for the four-tier design is about $112\times$ as large as that for the 2D design. However, the routing time for the four-tier design optimized by the gate composition algorithm is approximately $51\times$ as large as that for the 2D design. We find similar trends in almost all the other designs. Although smaller designs (e.g., 64-bit adders) not shown in the table obtain negligible benefits from the gate composition, all large designs (512- and higher-bit adders) clearly show the effectiveness of the gate composition algorithm for routing complexity reduction.

VI. CONCLUSION

Monolithic 3D integration is the most fine-grained integration technology, so design methodologies for the design of monolithic 3D ICs have actively been researched recently. Especially, simultaneous 3D routing effectively optimizes the wirelength of a given design. However, routing time increases significantly as the tier count goes up. In this paper, we proposed a gate composition algorithm to reduce the routing complexity. The simulation results show that the gate composition algorithm reduces routing time by up to $4.3\times$.

ACKNOWLEDGEMENT

This work was supported by the New Faculty Seed Grant 125679-002 funded by Washington State University.

REFERENCES

[1] C.-H. Shen, J.-M. Shieh, T.-T. Wu, W.-H. Huang, C.-C. Yang *et al.* Monolithic 3D Chip Integrated with 500ns NVM, 3ps Logic Circuits and SRAM. In *Proc. IEEE Int. Electron Devices Meeting*, pages 9.3.1–9.3.4, December 2013.

[2] C. Liu and S. K. Lim. A Design Tradeoff Study with Monolithic 3D Integration. In *Proc. Int. Symp. on Quality Electronic Design*, pages 529–536, March 2012.

[3] Yiting Chen and Dae Hyun Kim. A Legalization Algorithm for Multi-Tier Gate-Level Monolithic Three-Dimensional Integrated Circuits. In *Proc. Int. Symp. on Quality Electronic Design*, pages 277–282, 2017.

TABLE IV

RUNTIME FOR ROUTING (T-R) AND # MIVS IN EACH TIER OF THE 2D AND 3D DESIGNS. M-Tn DENOTES THE NUMBER OF MIVS IN TIER n.

Circuit	# Bits	2D	Two-tier 3D		Three-tier 3D			Four-tier 3D			
		T-R (s)	T-R (s)	M-T2	T-R (s)	M-T2	M-T3	T-R (s)	M-T2	M-T3	M-T4
Brent-Kung	64	1 (1.00)	30 (30.00)	204	37 (37.00)	240	140	53 (53.00)	226	177	124
	128	2 (1.00)	43 (21.50)	400	110 (55.00)	491	292	186 (93.00)	487	402	223
	256	5 (1.00)	71 (14.20)	857	262 (52.40)	954	614	284 (56.80)	890	735	446
	512	11 (1.00)	161 (14.64)	1,716	440 (40.00)	1,815	1,178	817 (74.27)	1,780	1,456	862
	1,024	30 (1.00)	381 (12.70)	3,616	2,347 (78.23)	3,861	3,127	5,258 (175.27)	3,643	3,350	1,937
Han-Carlson	64	2 (1.00)	28 (14.00)	273	116 (58.00)	307	207	168 (84.00)	296	266	165
	128	4 (1.00)	88 (22.00)	584	136 (34.00)	676	442	404 (101.00)	624	612	382
	256	8 (1.00)	197 (24.63)	1,289	476 (59.50)	1,385	1,072	1,301 (162.63)	1,231	1,232	843
	512	20 (1.00)	462 (23.10)	2,748	1,351 (67.55)	2,861	2,213	5,178 (258.90)	2,782	2,622	1,755
	1,024	53 (1.00)	1,175 (22.17)	5,779	5,447 (102.77)	6,171	5,018	13,523 (255.15)	5,580	5,566	4,044
Kogge-Stone	64	2 (1.00)	53 (26.50)	514	148 (74.00)	528	373	320 (160.00)	457	497	348
	128	6 (1.00)	104 (17.33)	1,217	432 (72.00)	1,126	942	2,095 (349.17)	1,030	1,147	756
	256	17 (1.00)	299 (17.59)	2,606	1,601 (94.18)	2,415	2,182	3,722 (218.94)	2,167	2,501	1,760
	512	41 (1.00)	618 (15.07)	5,809	3,642 (88.83)	5,817	4,897	7,282 (177.61)	4,698	5,266	3,812
	1,024	157 (1.00)	1,887 (12.02)	12,756	14,301 (91.09)	13,183	12,190	17,543 (111.74)	11,846	13,396	10,082
Ladner-Fischer	64	2 (1.00)	25 (12.50)	276	110 (55.00)	297	250	142 (71.00)	307	256	184
	128	4 (1.00)	92 (23.00)	661	209 (52.25)	627	495	319 (79.75)	590	550	388
	256	8 (1.00)	198 (24.75)	1,427	391 (48.88)	1,403	1,115	787 (98.38)	1,229	1,212	861
	512	19 (1.00)	447 (23.53)	2,974	1,189 (62.58)	2,887	2,381	5,003 (263.32)	2,553	2,520	1,721
	1,024	46 (1.00)	884 (19.22)	6,233	5,041 (109.59)	5,512	5,205	6,820 (148.26)	5,151	5,467	3,892

TABLE V

WIRELENGTH (WL) AND RUNTIME FOR ROUTING (TR). ALL THE VALUES ARE SCALED TO THE VALUES OF THE 2D DESIGNS.

Circuit	# Bits	2D		Without gate composition						With gate composition					
				Two-tier 3D		Three-tier 3D		Four-tier 3D		Two-tier 3D		Three-tier 3D		Four-tier 3D	
		WL	TR	WL	TR	WL	TR	WL	TR	WL	TR	WL	TR	WL	TR
Brent-Kung	512	1.00	1.00	0.852	14.64	0.771	40.00	0.719	74.27	0.845	10.55	0.751	33.18	0.707	53.36
	1,024	1.00	1.00	0.777	12.70	0.671	78.23	0.616	175.27	0.773	11.23	0.668	38.20	0.606	88.33
Han-Carlson	512	1.00	1.00	0.783	23.10	0.695	67.55	0.635	258.90	0.793	12.85	0.689	39.55	0.628	86.30
	1,024	1.00	1.00	0.771	22.17	0.665	102.77	0.593	255.15	0.772	8.43	0.666	31.81	0.588	75.34
Kogge-Stone	512	1.00	1.00	0.769	15.07	0.656	88.83	0.583	177.61	0.800	9.22	0.685	24.34	0.602	70.51
	1,024	1.00	1.00	0.735	12.02	0.605	91.09	0.584	111.74	0.765	8.17	0.640	35.65	0.606	52.48
Ladner-Fischer	512	1.00	1.00	0.876	23.53	0.844	62.58	0.809	263.32	0.889	12.58	0.809	36.00	0.758	61.05
	1,024	1.00	1.00	0.857	19.22	0.808	109.59	0.772	148.26	0.859	16.00	0.771	47.70	0.758	51.41

[4] D. H. Kim, K. Athikulwongse, and S. K. Lim. Study of Through-Silicon-Via Impact on the 3D Stacked IC Layout. In *IEEE Trans. on VLSI Systems*, volume 21, pages 862–874, May 2013.

[5] D. H. Kim, S. Mukhopadhyay, and S. K. Lim. TSV-Aware Interconnect Distribution Models for Prediction of Delay and Power Consumption of 3-D Stacked ICs. In *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, volume 33, pages 1384–1395, September 2014.

[6] D. Henry, X. Bailin, V. Lapras, MH. Vaudaine, JM. Quemper *et al.* Via First Technology Development Based on High Aspect Ratio Trenches Filled with Doped Polysilicon. In *IEEE Electronic Components and Technology Conf.*, pages 830–835, May 2007.

[7] J. U. Knickerbocker, P. S. Andry, B. Dang, R. R. Horton, M. J. Interrante *et al.* Three-Dimensional Silicon Integration. In *IBM Journal of Research and Development*, pages 553–569, November 2008.

[8] Bon Woong Ku, Kyungwook Chang, and Sung Kyu Lim. Compact-2D: A Physical Design Methodology to Build Commercial-Quality Face-to-Face-Bonded 3D ICs. In *Proc. Int. Symp. on Physical Design*, pages 90–97, 2018.

[9] Sheng-En David Lin, Partha Pratim Pande, and Dae Hyun Kim. Optimization of Dynamic Power Consumption in Multi-Tier Gate-Level Monolithic 3D ICs. In *Proc. Int. Symp. on Quality Electronic Design*, pages 29–34, 2016.

[10] M.-K. Hsu, Y.-W. Chang, and V. Balabanov. TSV-Aware Analytical Placement for 3D IC Designs. In *Proc. ACM Design Automation Conf.*, pages 664–669, June 2011.

[11] Nangate. Nangate 45nm Open Cell Library. http://www.nangate.com.

[12] NCSU. FreePDK45. http://www.eda.ncsu.edu/wiki/FreePDK.

[13] Shreepad Panth, Kambiz Samadi, Yang Du, and Sung Kyu Lim. Design and CAD Methodologies for Low Power Gate-level Monolithic 3D ICs. In *Proc. Int. Symp. on Low Power Electronics and Design*, pages 171–176, August 2014.

[14] Shreepad Panth, Kambiz Samadi, Yang Du, and Sung Kyu Lim. Shrunk-2D: A Physical Design Methodology to Build Commercial-Quality Monolithic 3D ICs. In *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, volume 36, pages 1716–1724, October 2017.

[15] S. Bobba, A. Chakraborty, O. Thomas, P. Batude, and G. de Micheli. Cell Transformations and Physical Design Techniques for 3D Monolithic Integrated Circuits. volume 9, pages 19:1–19:28, September 2013.

[16] S. Bobba, A. Chakraborty, O. Thomas, P. Batude, T. Ernst *et al.* CELONCEL: Effective Design Technique for 3-D Monolithic Integration Targeting High Performance Integrated Circuits. In *Proc. Asia and South Pacific Design Automation Conf.*, pages 336–343, January 2011.

[17] S. Panth, K. Samadi, Y. Du, and S. K. Lim. Design and CAD Methodologies for Low Power Gate-level Monolithic 3D ICs. In *Proc. Int. Symp. on Low Power Electronics and Design*, pages 171–176, August 2014.

[18] S. Panth, K. Samadi, Y. Du, and S. K. Lim. Placement-Driven Partitioning for Congestion Mitigation in Monolithic 3D IC Designs. In *Proc. Int. Symp. on Physical Design*, pages 47–54, March 2014.

[19] V. Sundaram, Q. Chen, T. Wang, H. Lu, Y. Suzuki *et al.* Low Cost, High Performance, and High Reliability 2.5D Silicon Interposer. In *IEEE Electronic Components and Technology Conf.*, pages 342–347, May 2013.

[20] Y.-J. Lee and S. K. Lim. Ultrahigh Density Logic Designs Using Monolithic 3-D Integration. In *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, volume 32, pages 1892–1905, December 2013.