An Iterative Partitioning Refinement Algorithm Based on Simulated Annealing for 3D VLSI Circuits

[1,2]Sandro Sawicki, [1]Gustavo Wilke, [1]Marcelo Johann, [1]Ricardo Reis
{sawicki, wilke, johann, reis}@inf.ufrgs.br

[1] UFRGS – Universidade Federal do Rio Grande do Sul
Instituto de Informática

[2] UNIJUI – Universidade Regional do Noroeste do Estado do Rio Grande do Sul
Departamento de Tecnologia

Abstract

Partitioning algorithms are responsible for dividing random logic cells and ip blocks into the different tiers of a 3D design. Cells partitioning also helps to reduce the complexity of the next steps of the physical synthesis (placement and routing). In spite of the importance of the cell partitioning for the automatic synthesis of 3D designs it been performed in the same way as in 2D designs. Graph partitioning algorithms are used to divide the cells into the different tiers without accounting for any tier location information. Due to the single dimensional alignment of the tiers connections between the bottom and top tiers have to go through all the tiers in between, e. g., in a design with 5 tiers a connection between the top and the bottom tiers would require 4 3D vias. 3D vias are costly in terms of routing resources and delay and therefore must be minimized. This paper presents a methodology for reducing the number of 3D vias during the circuit partitioning step by avoiding connections between non-adjacent tiers. The proposed algorithm minimizes the total number of 3D vias and long 3D vias while respecting area balance, number of tiers and I/O pins balance. Experimental results show that the number of 3D-Vias was reduced by 19%, 17%, 12% and 16% when benchmark circuits were designed using two, three, four and five tires.

1. Introduction

The design of 3D circuits is becoming a reality in the VLSI industry and academia. While the most recent manufacturing technologies introduces many wire related issues due to process shrinking (such as signal integrity, power, delay and manufacturability), the 3D technology seems to significantly aid the reduction of wire lengths [1-3] consequently reducing these problems. However, the 3D technology also introduces its own issues. One of them is the thermal dissipation problem, which is well studied at the floorplanning level [4] as well as in placement level [3]. Another important issue introduced by 3D circuits is how to address the insertion of the inter-tier communication mechanism, called 3D-Via, since introduces significant limitations to 3D VLSI design. This problem has not been properly addressed so far since there some aspects of the 3D via insertion problem that seem to be ignored by the literature: 1) all face-to-back integration of tiers imply that the communication elements occupy active area, limiting the placement of active cells/blocks ; 2) the 3D-Via density is considerably small compared to regular vias, which won't allow as many vertical connections as could be desired by EDA tools; 3) timing of those elements can be bad specially considering that a vertical connection needs to cross all metal layers in order to get to the other tier ; 4) 3D-Vias impose yield and electrical problems not only because of their recent and complex manufacture process but also because they consume extra routing resources.

The 3D integration can happen in many granularity levels, ranging from transistor level to core level. While core level and block level integration are well accepted in the literature, there seem to exist some resistance to the idea of placing cells in 3D [6]. One of the reasons is that finer granularity demands higher 3D-Via demand, which might fail to meet the physical constraints imposed by them. On the other hand, the evolution of the via size is going fast and is already viable (for most designs) to perform such integration [2, 5, 11, 13] since we already can build 0.5 μm pitch face-to-face vias [6] and 2.4 μm pitch on face-to-back [5]; we believe that this limitation is more in the design tools side, since those are still not ready to cope with the many issues of 3D-Vias [7, 13].

While it is known that the addition of more 3D-Vias improves wire length [5], this design methodology might fail to solve the issues highlighted above. We believe that EDA tools can play a major role to enable random logic in 3D by minimizing 3D-Vias to acceptable levels. The number of 3D-Vias required in a design is determined by the tier assignment of each cell, which is performed during the cell partitioning. The cell partitioning is usually performed by hypergraph partitioning tools (since it is straightforward to map a netlist into a hypergraph) such as hMetis [8] as done in [2] for the same purpose that is addressed here. On the other hand, hypergraph tools do not understand the distribution of partitions in the space (in 3D circuits they are distributed along in a single dimension) and fail to provide optimal results. It is important to understand that the amount of resources used is proportional to of the vertical distance of the tiers; in fact, considering that the path from a tier to an adjacent involves regular vias going through all metal layers plus one 3D-Via, it is

clear that any vertical connection larger than adjacency might be too expensive in terms of routing resources and delay.

In this paper we want to demonstrate that using a Simulated Annealing based algorithm we can refine the partitioning produced by traditional hypergraph partitioning algorithms further minimize the 3D-Via count while maintain nets vertically shorter and keeping good I/O and area balancing. The remainder of the paper is organized as follow. In Section 2 presents the problem being addressed, section 3 presents how we draft a solution to this problem while comparing it to similar approaches. Section 4 presents the details of our Simulated Annealing algorithm and finally section 5 presents conclusions and directions for the future work.

2. Problem Formulation

Consider a random logic circuit netlist and a target 3D circuit floorplan (including area and number of tiers), compute the partitioning of the I/O pins as well as the partitioning of cells into tiers such that the 3D-Vias count is minimized; be constrained by keeping a reasonable balance of both, I/Os and cells, along the tiers.

3. Partitioning Algorithm and Related Work

Our previous work [9] presents a solution for the proposed problem. In that work, we concentrated solely in improving the I/O pins partitioning and the cells partitioning were naturally following the improved structure producing cut improvements in the order of 5,33% (2 tiers), 8,29 (3 tiers), 9,59% (4 tiers) and 16,53% (5 tiers). We have used hMetis for cell partitioning the while the I/Os were fixed by our method. Experimental results have shown to that the initial I/O placement improves the ability of hMetis to partition cells. In this paper, we will tackle the cells partitioning method as well.

We now propose an iterative improvement heuristic to handle the proposed problem (section 2). The algorithm is inspired on Simulated Annealing, but instead of accepting uphill solutions to avoid local minima [14], our heuristic understands that we can obtain a good initial solution (sufficiently close to the optimal point) using our previous work and now focus on improving it disregarding any possible local minima.

The main difference between our new approach and a hypergraph partitioner such as hMetis is that our approach accounts for the location of the partitions. In fact, in a 3D circuit, the partitions are organized in a line, which implies in the notion of adjacent partitions (that are cheap in terms of cut) and distant partitions (that are expensive since demand extra 3D-Vias). Having the goal of minimizing the 3D-Vias as a whole, we naturally want to overpenalize the cut of distant partitions, which is not done in standard hypergraph partitions. For example, the algorithm in [2, 9] employs hMetis to partition the cells into groups, and in a second stage employs a post-processing method to distribute the partitions in the 1D space (line) such that the number of 3D-Vias is minimized (as illustrated in figure 1.a). While this approach targets the same goals, it clearly has limited freedom since partitions cannot be changed. The algorithm proposed in this paper allows cells to move from one partition to the other as long as the final cost is reduced, as illustrated in figure 1.b. The definition of the cost function is responsible for keeping a good I/O pins and cells balance among the different tiers.

4. Improvement Method for Heuristic Partitioning

The proposed algorithm picks an initial solution (the solution obtained from our previous work [9]) and improves it iteratively using random perturbations of the existing solution. The perturbations might be accepted or rejected depending on the cost variation. Any perturbation that improves the current state is accepted and all perturbations that increase the cost is rejected (greedy behavior).

4.1. Pertubation Procedure

The perturbation function designed for our application attempts to move cells across partitions. Although they are random in nature, we perform two different kinds of perturbations for better diversity: single movement or double exchange (swap). The double and single perturbations are alternated with 50% probability. They work as follows:

- The single perturbation can randomly pick a cell or an I/O pin (with 50% probability each) and move it to a different tier (also chosen randomly).
- The double perturbation randomly selects a pair of elements to switch partitions. Each element can be either a cell or I/O pin with 50% of selecting each, totaling 4 different double perturbation combinations, each having 25% probability of happening.

4.2. Cost Function

Any intermediate state of the partitioning process can have its quality measured by a cost function. In the cost function, we model all metrics of interest in a single value that represents the cost.

Our cost function is divided in three distinct parts: a cost v associated to the usage of 3D-Via resources, a value a for the area balance and finally a cost p for the I/O pin balance. The cost reported is a combination of the three parcels; in order to be able to add them together, we normalize each parcel by dividing them from their initial values vi, ai and pi (computed before the first perturbation). In addition, we also impose weights

(wv, wa and wp) in order to be able to fine tune the cost function to vias the optimization process, as shown in equation 1.

The values v, a and p are computed as follows.

- For each net, compute the square of the via count; add the computed number of each net to obtain v. The square is applied to highly punish nets having high 3D-Via counts and to encourage short vertical connections.
- To compute a, we first compute the cell area of all tiers; the unbalance cost is a subtraction of the largest by the smallest area.
- The value p is computed similarly to a.

$$c = \frac{(w_v \times v)}{v_i} + \frac{(w_a \times a)}{a_1} + \frac{(w_p \times p)}{p_1} \quad (1)$$

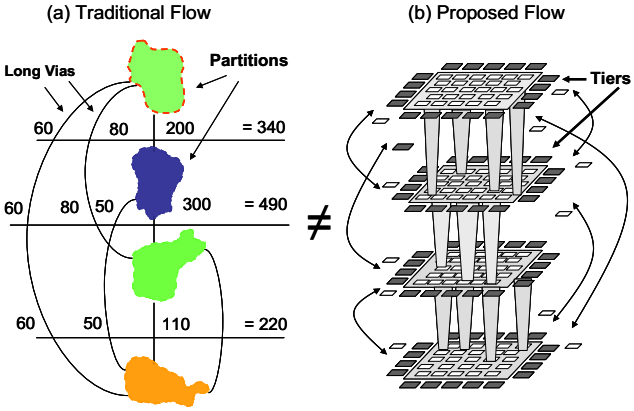

Figure 1. Fixed Tiers Method

5. Experimental Results

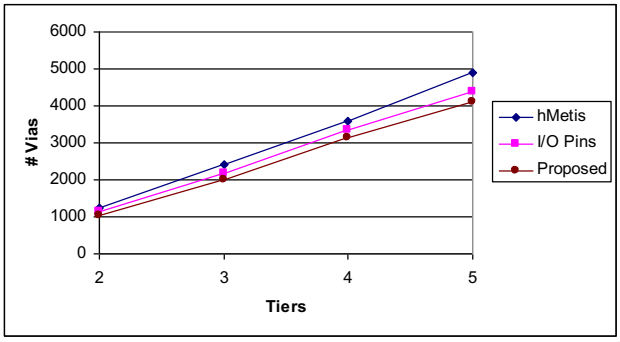

Figure 2. Number of 3D-Vias

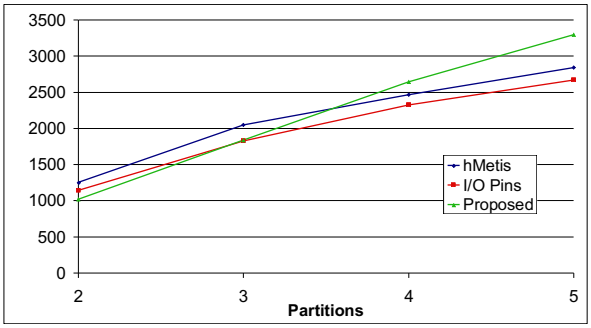

Figure 3. Cut quality

Our partition refinemt algorithm was compared to a state-of-the-art hypergraph partitioner called hMetis. We have used hMetis to partition the design netlist (including cells and I/O pins) freely into *n* partitions, where *n* is also the number of tiers. In a subsequent stage, we would assign the partitions into tiers, as illustrated in figure 1 inspired in the method performed by [2].Our previous work [9] would perform a slightly different approach; it would first partition the I/O pins of the block in a way that the I/O balance could be controlled; also, we have a heuristic that is able to cleverly partition the I/Os in a way that they can aid the cell partitioning to reduce the cut. Following the I/O partitioning, we fix the pins and perform hMetis to partition the cells. Since the method is concentrated on the I/Os, we now call it I/O Pins. The method presented here is referred as proposed in the tables and graphs below. Please note that this method starts from the I/O Pins solution and runs our greedy improvement heuristic to refine both the cells and the I/O partitioning.

The experimental setup is as follows. We picked circuits from the ISPD 2004 benchmark suite [12] and partitioned the design into 2, 3, 4 and 5 tiers. The three referred methods (hMetis, I/O Pins and proposed) were attempted. All methods were constrained to distribute area evenly, which resulted in a worst case of 0.1% unbalance. The I/O balancing is not imposed in the hMetis since it would overconstraint the method. For this reason, hMetis has the worst I/O balancing while the I/O Pins is the best since the proposed method uses a little freedom on the I/O balancing to improve the 3D-Via count.

Figure 2 shows the total 3D-Via count comparison between the methods. The proposed method obtains the average least amount of 3D-Vias. More specifically, the proposed method produced 3D-Via count average

improvement of 19% and 11% compared to hMetis and I/O Pins respectively for 2 tiers, 17% and 8% for 3 tiers, 12% and 6% for four and finally 16% and 7% for 5 tiers. Figure 3 shows the final partitioning from a hypergraph perspective. The y-axys shows the average cut among the different partitions when the circuits are divided into 2, 3, 4 and 5. It can be observed that the proposed algorithm presents a higher cut value when the number of partitions is larger, however when only two partitions are created the proposed algorithm achieves the smaller cut value than hMetis and I/O pins. The behavior is explained by the cost function used to optimized the final partition set. The proposed algorithm, unlike hMetis and I/O Pins, do not tackle at the cut reduction but in reduction of the total number of vias. When only two partitions are considered the number of vias given by the cut of the partitioning algorithm. When more partitions are created the proposed algorithm increases the number of connections between adjacent partitions in order to reduce the number of connections between non-adjacent ones. This behavior leads to a increase in the partitioning cut number while the total number of vias can be further reduced.

6. Conclusions

This paper presented a method to refine the I/O Pins and cells partitioning into a 3D VLSI circuit. We were able to develop a heuristic iterative improvement algorithm that achieves better partitioning than a simple application of hypergraph partitioners. The method demonstrates that hypergraph partitioners are not well suited for cell and I/O partitioning into 3D circuit because they do not handle long connections properly, affecting the total 3D-Via count. We demonstrated that our heuristic was able to improve the 3D-Via count by considering the positions of each tier within the 3D chip while partitioning the cells among the tiers. Finally, we highlight that our heuristic was able to perform partitioning while keeping area and I/O pin count balanced across tiers.

7. References

[1] W. R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A. M. Sule, M. Steer and P. D. Franzon; Demystifying 3D ICs: The Pros and Cons of Going Vertical. IEEE Design and Test of Computers – special issue on 3D integration; pp 498-510, Nov.-Dec. 2005.

[2] C. Ababei, Y. Feng, B. Goplen, H. Mogal, T. Zhang, K. Bazargan and S. Sapatnekar. Placement and Routing in 3D Integrated Circuits. IEEE Design and Test of Computers – special issue on 3D integration; pp 520-531, Nov.-Dec. 2005.

[3] B. Goplen; S. Sapatnekar; Efficient Thermal Placement of Standard Cells in 3D ICs using Forced Directed Approach. In: Internation Conference on Computer Aided Design, ICCAD'03, November, San Jose, California, USA, 2003.

[4] E. Wong; S. Lim. 3D Floorplanning with Thermal Vias. In: DATE '06: Proceedings of the Conference on Design, Automation and Test in Europe, 2006. p.878–883.

[5] Das, S.; Fan, A.; Chen, K.-N.; Tan, C. S.; Checka, N.; Reif, R. Technology, performance, and computer-aided design of three-dimensional integrated circuits. In: ISPD'04: Proceedings Of The 2004 International Symposium On 59 Physical Design, 2004, New York, NY, USA. Anais. . . ACM Press, 2004. p.108–115.

[6] Patti, R. Three-dimensional integrated circuits and the future of system-on-chip designs. Proceedings of IEEE, [S.l.], v.94, p.1214–1224, 2006.

[7] Hentschke, R. et al. 3D-Vias Aware Quadratic Placement for 3D VLSI Circuits. In: IEEE Computer Society Anual Symposium on VLSI, ISVLSI, 2007, Porto Alegre, RS, Brazil. Proceedings. . . Los Alamitos: IEEE Computer Society, 2007. p.67–72.

[8] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel Hypergraph Partitioning: Application in VLSI domain. In Proceedings of 34th Annual Conference on. Design Automation, DAC 1997, pages 526–529, 1997.

[9] Sawicki, S.; Hentschke, Renato ; Johann, Marcelo ; Reis, Ricardo . An Algorithm for I/O Pins Partitioning Targeting 3D VLSI Integrated Circuits. In: 49th IEEE International Midwest Symposium on Circuits and Systems, 2006, Porto Rico. MWSCAS, 2006.

[10] K. Bernstein; P. Andry; J. Cann; P. Emma; D. Greenberg; W. Haensch; M. Ignatowski; S. Koester; J. Magerlein; R. Puri; A. Young. Interconnects in the Third Dimension: Design Challenges for 3D ICs. In: Design Automation Conference, 2007. DAC'07. 44th ACM/IEEE. 2007 Page(s):562 - 567

[11] K. Banerjee and S. Souri and P. Kapur and K. Saraswat. 3D-ICs: A Novel Chip Design for Improving Deep Submicrometer Interconnect Performance and Systems on-Chip Integration. Proceedings of IEEE, vol 89, issue 5, 2001.

[12] ISPD 2004 - IBM Standard Cell Benchmarks with Pads. http:// www. public. Iastate .edu/~nataraj/ISPD04_Bench.html#Benchmark_Description. Access on Mar 2008.

[13] R. Hentschke, G. Flach, F. Pinto, and R. Reis, "Quadratic Placement for 3D Circuits Using Z-Cell Shifting, 3D Iterative Refinement and Simulated Annealing," Proc. Symp. on Integrated Circuits and Syst. Des. '06, 220-225.

[14] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, Optimization by simulated annealing, Science, 1983, 220, pages 671-680.