Unbalacing the I/O Pins Partitioning for Minimizing Inter-Tier Vias in 3D VLSI Circuits

Sandro Sawicki[1,2] sawicki@inf.ufrgs.br
Renato Hentschke[1] renato@inf.ufrgs.br
Marcelo Johann[1] johann@inf.ufrgs.br
Ricardo Reis[1] reis@inf.ufrgs.br

[1] UFRGS – Universidade Federal do Rio Grande do Sul
PPGC - Instituto de Informática
Porto Alegre, Brazil

***Abstract*—The 3D Circuit technologies appear as a possible solution for interconnect optimization. For most of the 3D technologies, the 3D-Vias represent a very complex issue because of large pitch requirements and heavy usage of routing constraints. New algorithms and CAD methods must be developed in order to take advantage of the high integration of elements and potentially shorter wire lengths while keeping track of the 3D-Vias. One of the CAD problems, addressed by this paper, is the partition and placement of the I/O pins of a block into sub-blocks that are partitioned into the circuit tiers. In this paper, we extend our previous work in the field to trade-off the I/O pins balance for improved cut. The new version of our partitioning algorithm outperformed the widely used hMetis algorithm in number of 3D-Vias from 2% to 10% (depending on the I/O pins balance), while the standard deviation of the number of I/Os increased. We also observed that the maximum number of 3D-Vias between pairs of adjacent tiers dropped by 14% (in the best case) with the I/O pin unbalance.**

I. INTRODUCTION

One of the recently proposed solutions for interconnect optimization is 3D circuits design [4, 5]. There is a massive interest from both industry and academia in 3D integration issues. Major companies like IBM, Intel, Samsung, Micron, Cadence and Infineon are some examples.

A 3D circuit is made of the stacking of regular circuits forming active islands known as tiers [5]. Between the tiers there are metal layers and insulator layers. The communication between tiers is made with the so called 3D-Via. Among the existing technologies for the fabrication of the 3D-Vias there is a wide range of pitch requirements. According to [5], the ones that provide the highest integration are the *face-to-face* technologies and *through Vias* in a *face-to-back* fashion. Although they are relatively small, all of them consume routing resources on all metal layers and particularly the *face-to-back* 3D-Vias occupy active area. For all these reasons and for their possible high resistance and capacitance, the 3D-Vias are one of the most important issues to be considered by the 3D CAD community.

The research on CAD for 3D is still starting to appear**.** The problem of 3D placement is different from its original definition to account for three dimensions wirelength optimization [8, 12], thermal improvement during placement [1, 7] and 3D-Vias minimization using min-cut partitioning [2, 5, 6]. The most usual placement flow in the literature is to start partitioning the design into several sub-circuits minimizing the total cut (number of 3D-Vias). According to [9] and [5], the 3D-Vias in Bulk technologies highly penalizes the active area of a 3D placement solution. This experiment strongly motivates the 3D-Via minimization flow when targeting this kind of technology.

The quadratic placement [3] algorithm is one of the most successful approaches for 2D circuits placement, while for 3D circuits is starting to be studied [7, 8]. One of the requirements of these algorithms is to have I/O pins in the boundary of the chip. For this reason, an algorithm for the I/O pins partitioning was recently proposed [9, 14]. Its main usage is to distribute the I/Os in the various tiers of the design in such a way that the I/Os are able to facilitate the cells partitioning with the objective of minimizing the cut (number of 3D-Vias). The previous experimental results presented in [9,14] report an average 3D-Via minimization improvement of 2-3% compared to hMetis that is largely used for min-cut partitioning and have been used by many researchers on 3D placement for minimizing the number of 3D-Vias. We observe that, for our application, reducing the cut is mandatory in order to implement a 3D circuit with Bulk technology for instance. Any improvement has an important impact on the quality of the circuit. Additionally, using hMetis for the whole netlist in a single step leads to prohibitively unbalance of the I/Os [9,14].

Our partitioning flow is based on the logic distance between I/Os pins in a two phase partitioning method: first

[2] On live from UNIJUI, Northwest University of Rio Grande do Sul, DETEC – Departamento de Tecnologia, Santa Rosa, Brazil.

* This research was partially supported by a grant from CNPq, Brazil.

the I/Os are modeled as a complete graph using the logic distance as cost of the connections; the graph is then partitioned using min-cut partitioning. The second stage is to fix the I/Os and partition the rest of the netlist. The details of this partitioning method are reviewed in section II.

In this paper we study the proposed netlist partitioning method targeting a deeper minimization of the 3D-Vias. We focus our study on methods for partitioning the I/Os in a more flexible way in order to provide a better starting point for the subsequent I/O partitioning problem. We do not change the cells partitioning method. Section II reviews the I/O partitioning approach while pointing out the room for improvement that is explored in this paper. Section III presents the experimental results and conclusions are given in section IV.

II. The I/O Partitioning Problem and The Proposed Modifications in the Algorithm

The I/O partitioning problem can be summarized as follows. The set of I/O pins P of a regular (2D) netlist must be partitioned into t tiers to the subsets P_1, P_2, ... P_t. After that, the partitions must be mapped to a tier. We call this as *tier assignment* problem. Finally, the I/O pins must be placed into their assigned tier. We constraint the problem in order to preserve the ordering and orientation of the original I/O pins placement. Figure 1 illustrates the I/O partitioning problem, that we call *netlist migration*.

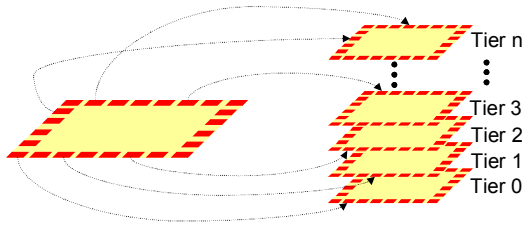

Figure 1. Migration from 2D ICs to 3D ICs

Our I/O partitioning algorithm is a heuristic to be combined with existing min-cut partitioning approaches. We perform the I/O partition in two steps: first, a complete graph of the I/O pins is created with costs associated to each edge; second, a min-cut partitioning is performed considering the calculated costs. The following steps of the algorithm will calculate the area of each tier and the consequent I/O placement.

After I/O partitioning and placement, we perform a min-cut partitioning of the cells fixing the I/Os. At this point, a good partitioning of the I/Os could facilitate the cells partitioning since the I/Os are the beginning and end points of paths in the netlist. Our logic distance metric in the first phase provides an insight of the netlist structure for the partitioning algorithm since it contains information of the whole netlist synthesized in cost numbers.

In previous work, we compared the algorithm with two other approaches. The first, called **unlocked_pins**, is preformed by hMetis [10,13] partitioning the whole netlist (pins + cells). This is the solution adopted by most of the 3D placers in the literature, such as [1, 2, 6]. This approach was used by us as a *good in terms of cut* (number of 3D-Vias) but not in terms of I/O balanced distribution. The second approach is called **alternate_pins**. It consists of partitioning the I/Os using their placement information and assigning one by one alternatively to a partition. It was used as *good for I/O balance* but not for number of 3D-Vias. The experimental results presented in [9,14] show that the proposed approach achieves a very good balance that is close to the one obtained by the alternate_pins method along with a number of 3D-Vias minimization that outperformed both algorithms. This fact can be explained by our pre-processing stage that computes the logic distance between I/Os requiring intensive CPU usage. The distances are stored in a file so that the I/O partitioning runtimes are not harmed.

In summary, our method turned out to be good for both balance and number of 3D-Vias. Targeting technology in which the 3D-Vias costs too much (as detailed in [5,9]), the 3D-Via minimization is mandatory and our method is able to further minimize the 3D-Vias compared to existing approaches in the literature [1, 2, 5,6].

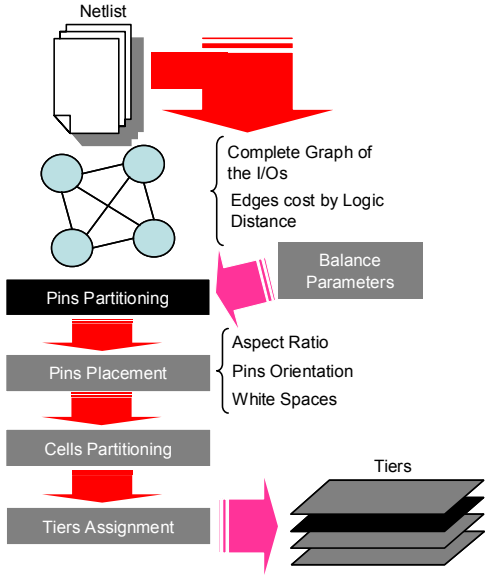

Figure 2. An Illustration of Our Flow

The figure 2 summarizes the flow of the I/O partitioning algorithm. We highlight the step of *Pins Partitioning*. This step is performed by the hMetis algorithm [13]. At this point, we can control the balance of the I/O pins. In the previous work, we imposed a tight balance in order to achieve a comparable I/O distribution with the **alternate_pins** method. However, in this work, we are focusing on achieving an even smaller cut (number of 3D-Vias) by relaxing the pin balance constraint. It is well known that a tight balance constraint over-constraints the partitioning process [13]. However, it is really not clear whether an unbalanced I/O Pin solution would be a better starting point for the cell partitioning. In this paper we investigate this matter keeping the rest of the flow with a tight constraint. The subsequent cell partitioning

works with the I/Os as fixed pins and is constrained by a very tight balance for the cells area.

In our algorithm the parameter used to specify the unbalance between the partitions during recursive bisection is an integer number between 1 and 49 (same as the hMetis tool). For example, considering a hypergraph with n vertices, each having a unit weight, let u be the unbalance parameter. Then, suppose the number of desired partitions is two, the number of vertices assigned to each one will be calculated according to the following equation:

$$\frac{(50-u)\times n}{100};\frac{(50+u)\times n}{100}$$

For example, let u = 10, then the bisection balance will range from 40%-60% to 60%-40%. Now suppose that we have four partitions, then an unbalancing factor 10 will result in partitions that can contain between $0.40^2n = 0.15n$ and $0.60^2n = 0.35n$ vertices.

III. Experimental Results

We have conducted experiments in order to verify the effectiveness of our approach. The benchmark set used is the ISPD 2004 placement benchmarks [11]. We started the experiments be recovering the experimental results of our algorithm from previous works [9,14] in which the balance criteria u is maximum (in our case, u = 1) presented in table 1. We than modified the balance to u = 10 and u = 25.

TABLE I. Comparison of the #3D-vias considering differents algorithms (recovered from [9,14])

Algorithms	2 tiers	3 tiers	4 tiers	5 tiers
unlocked_pins #vias	882	1,951	3,305	4,435
alternate_pins #vias	882	1,983	3,334	4,588
our_algorithm #vias	859	1,888	3,209	4,455

The results are reported on table 1. We report the effects on the I/O balance measured by the standard deviation of the number of pins. Suppose that a circuit is implemented with three tiers; the number of 3D-Vias between the first pair of adjacent tiers is 50 and the second pair has 60 vias. In this case, the standard deviation is 7.07. In table 2 we also report the total number of 3D-Vias and finally the worst case in number of 3D-Vias between a pair of adjacent tiers. In average, we can observe that the number improved from 3% (4 tiers) to 14% (3 tiers). Figure 3 shows two graphs reporting the average evolution on the number of Vias (total (a) and worst case of adjacent tiers (b)). It can be observed that the total number of 3D-Vias can reduce up to 7% while the number of 3D-Vias between pairs of tiers can reduce up to 14% with the unbalance of number of tiers. The standard deviation of I/Os ranges from 1% to 35% of the total number of I/Os. We highlight that the worst case obtained standard deviation is still much better than the one obtained by hMetis, as reported in [9, 13], that ranges from 39% to 57% and some tiers have no I/Os.

IV. Conclusions

This paper presented an evolution of an I/O pin algorithm targeting 3D circuits. We present an automated method for the migration of a 2D netlist for 3D. The I/Os are partitioned and placed in the boundary the new area planned for the 3D placement space. In previous works, we studied the effect of a smart I/O partitioning method and how it is able to improve the number of 3D-Vias indirectly, since it is only the first step of the actual cells partitioning. We reported average gains in the order of 2-3% in the number of 3D-Vias compared to hMetis partitioning for the whole netlist in a single step. Additionally, we have shown in previous work that such method is not feasible due to high unbalance of the I/O distribution. It is important to notice that most of the existing literature on 3D Placers that are based on hMetis for minimizing the number of 3D-Vias such as [1, 2, 5, 6].

In this paper, we investigated ways to further minimize the cut. We studied how the unbalance of the number of I/Os could help the subsequent cells partitioning process. We relaxed the I/O pin balance constraint keeping the area evenly distributed since the second partitioning process is still highly constrained. Adding up the advantage reported in previous works with the improvements achieved on this paper, we can outperform hMetis from 2% to 10% in average. This advantage can be explained by the fact that the logic distance provides a good insight of the netlist structure, facilitating the search.

References

[1] 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.

[2] C. Ababei; H. Mogal; K. Bazargan; Three-Dimensional Place and Route for FPGAs. *In:* Proceedings of the Design Automation Conference - Asia and South Pacific*,* ASP-DAC 2005. Volume: 2 18-21 Jan. 2005. Page(s): 773- 778 Vol. 2.

[3] C. Alpert; T. Chan; D. J. Huang.; I. Markov; K. Yan. Quadratic Placement Revisited. *In: Proc. of the 34th Annual Conference on Design Automation,* DAC 1997, New York, NY, USA. Anais. . . ACM Press, 1997. p.752–757.

[4] 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.

[5] 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.

[6] Y Deng; W. Maly. Interconnect Characteristics of 2.5-D System Integration Scheme*. In: Proceedings of the International Symposium on Physical Design,* ISPD 2001, New York, NY, USA. Anais. . . ACM Press, 2001. p.171– 175.

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

[8] R. Hentschke, G. Flach, F. Pinto, R. Reis. Quadratic Placement for 3D Circuits Using Z-Cell Shifting, 3D Iterative Refinement and Simulated Annealing. *In 19th Symposium on Integrated Circuit and System Design,* (in Press) SBCCI 2006, Ouro Preto, Brazil.

[9] R. Hentschke, S. Sawicki, M. Johann, R. Reis. An Algorithm for I/O Partitioning Targeting 3D Circuits and Its Impact on 3D-Vias, *In IFIP Internation Conference on Very Large Scale Integration*, (unpublished- submitted) VLSI-SOC 2006.

[10] Hypergraph & Circuit Partitioning at hMetis Home Page, http://glaros.dtc.umn.edu/gkhome/views/metis/hmetis/. Access on Mar 2006.

[11] ISPD 2004 - IBM Standard Cell Benchmarks with Pads. http://www.public.iastate.edu/~nataraj/ISPD04_Bench.html#Benchmark_Description. Access on Mar 2006.

[12] S. Obenaus, T. Szymanski. Gravity: Fast Placement for 3D VLSI. ACM *Transactions on Design Automation of Electronic Systems*, New York, v.8, p.69–79, March 1999.

[13] 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.

[14] S. Sawicki, R. Hentschke, M. Johann, R. Reis. An Algorithm for I/O Pins Partitioning Targeting 3D VLSI Integrated Circuits. *In: 49th IEEE International Midwest Symposium on Circuits and Systems,* (in Press) MWSCAS 2006, Puerto Rico.

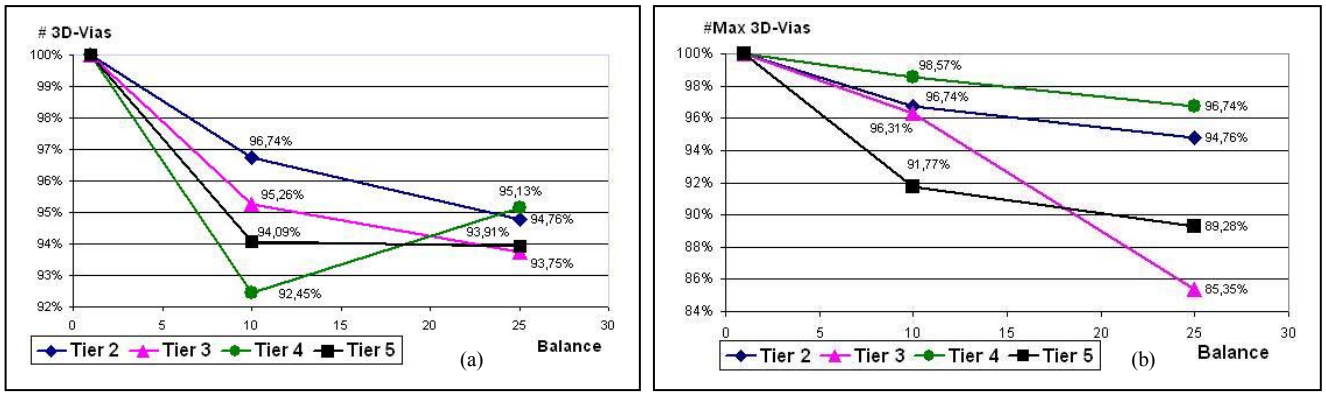

Figure 3. Evolution of the number of 3D-Vias obtained with the unbalance of the I/O paritioning process.

TABLE II. COMPARISON OF THE 3D-VIAS AND #MAX 3DVIAS CONSIDERING DIFFERENTS BALANCE PARAMETERS

ibm01	2 Tiers			3Tiers			4 Tiers			5 Tiers		
Balance	#vias Total	σ I/O	#max Vias	#vias Total	σ I/O	#max Vias	#vias Total	σ I/O	#max Vias	#vias Total	σ I/O	#max Vias
1	393	4	393	577	6	323	945	4	394	1,278	4	472
10	386	35	386	544	29	**339**	862	25	369	1,298	27	465
25	383	88	383	536	78	**379**	926	56	393	1,271	57	428
ibm02	**2 Tiers**			**3 Tiers**			**4 Tiers**			**5 Tiers**		
1	477	5	477	915	4	540	1,365	4	545	2,052	3	767
10	413	37	413	782	31	440	1,339	26	509	1,980	29	565
25	500	93	**500**	770	82	466	1,395	66	**557**	1,900	60	731
ibm03	**2 Tiers**			**3 Tiers**			**4 Tiers**			**5 Tiers**		
1	1,103	5	1,103	2,473	4	1,365	3,391	4	1,423	5,257	5	1,683
10	1,051	40	1,051	2,495	47	1,301	3,215	24	1,360	5,143	31	1,663
25	997	101	997	2,360	90	1,249	3,355	64	**1,451**	5,098	65	1,592
ibm04	**2 Tiers**			**3 Tiers**			**4 Tiers**			**5 Tiers**		
1	733	5	733	1,720	5	933	2,955	4	1,203	3,646	4	1,124
10	724	42	724	1,687	34	**964**	2,836	29	**1,211**	3,323	31	996
25	683	103	683	1,548	91	855	2,780	58	1,174	3,494	66	1,045
ibm06	**2 Tiers**			**3 Tiers**			**4 Tiers**			**5 Tiers**		
1	1,059	3	1,059	2,100	3	1,065	4,544	3	1,725	6,031	3	1,732
10	1,054	24	1,054	2,057	20	1,033	4,131	17	1,695	5,764	19	1,678
25	1,051	57	1,051	2,037	41	1,048	4,416	38	1,646	5,676	36	1,650
ibm07	**2 Tiers**			**3 Tiers**			**4 Tiers**			**5 Tiers**		
1	1,032	5	1,032	2,286	5	1,414	3,960	3	1,718	5,755	5	1,933
10	1,017	42	1,017	2,182	24	1,378	3,032	29	1,674	5,375	31	1,789
25	956	103	956	2,067	83	1,313	3,726	58	1,641	5,325	66	1,698
ibm08	**2 Tiers**			**3 Tiers**			**4 Tiers**			**5 Tiers**		
1	1,297	4	1,297	3,241	4	1,745	5,407	3	1,907	6,849	3	2,103
10	1,274	24	1,274	3,044	34	1,593	5,361	26	1,873	6,800	34	2,155
25	1,242	86	1,242	3,103	78	1,660	4,792	57	1,680	6,675	56	1,949
ibm09	**2 Tiers**			**3 Tiers**			**4 Tiers**			**5 Tiers**		
1	778	5	778	1,853	4	1,959	3,103	4	1,158	4,769	4	1,759
10	728	42	728	1,848	34	1,948	2,956	29	**1,216**	3,854	26	1,194
25	696	101	696	1,836	91	1,004	3,032	65	1,138	3,757	61	1,245
Average	**2 Tiers**			**3 Tiers**			**4 Tiers**			**5 Tiers**		
1	859	5	859	1,896	4	1,168	3,209	4	1,259	4,455	4	1,447
10	831	36	831	1,830	32	1,125	2,967	26	1,238	4,192	29	1,313
25	814	92	814	1,782	79	997	3,053	58	1,210	4,150	58	1,292