A method to speed up VLSI hierarchical physical design in floorplanning

Yanling Zhou[1], Yunyao Yan [2], Wei Yan[3]*

[1] School of Software&Microelectronics, Peking University, Beijing 100871, China
[2] School of Electronic and Information Engineering, Beijing JiaoTong University, Beijing 100044, China
[3] School of Software&Microelectronics, Peking University, Beijing 100871, China
* Email:yanwei@ss.pku.edu.cn

Abstract — *With the rapid increase in size and complexity of VLSI, it is hard to meet speed and quality requirement of IC physical design. In this paper, we present an efficient model for quick floorplanning in VLSI top-down hierarchical physical design flow using the Active-Logic Reduction Technology. The simplified model replaces some original modules in netlist file with filling units which have no logical connections. This method can effectively reduce internal logical units and quickly predict if chip design achieves timing closure after top and blocks implementation with this floorplan so as to quickly judge the floorplan's quality. Most importantly, it can maintain design quality while speeding up design flow. The results of six experiments show that the method can drastically reduce runtime by 6.2 times and memory by 2.8 times on average in VLSI hierarchical physical designs.*

Index Terms — VLSI, Hierarchical physical design, Floorplanning, Active-Logic Reduction Technology

1. Introduction

The IC physical design is a process of gate-level netlist synthesis, floorplanning, power planning, placement, preroute (mainly is trail route), CTS, routing and finally creating GDSII layout file. Floorplanning is basic in physical design, it decides the results of placement and routing (P&R) [1]. With the advanced manufacturing process, increasing chip integration and smaller size, also with the appearance of Nano scale VLSI design [2], VLSI design becomes increasingly dependent upon EDA tools with high quality and efficiency. In floorplanning stage, the complex netlist generally contains tens of millions gate-level units, R.Otten proposed automatic floorplan design method in 1982 [3]. It is difficult for EDA tools to complete floorplanning and P&R quickly and accurately, so EDA tools need several constraints for floorplanning. Through multiple iterations and repeated adjustment of floorplaning according to the results of P&R, to finally finish physical design and meet design requirements [1]. Besides, many new algorithms and techniques are applied to VLSI floorplanning, such as simulated annealing algorithm in [4], guided incremental floorplan algorithm to efficiently reduce the IR-drop violations with the B*-tree representation in [5] and another P/G network and floorplan method for fast design convergence in [6]. However, it is inefficient to spend a lot of time with repeated iterations adjustment, thus it is urgent to find a way to reduce floorplanning or evaluation runtime using advanced algorithms and model optimization to speed up ASIC physical design, but also maintain floorplanning's quality [6][7].

2. Quick floorplanning method

2.1 Physical design process

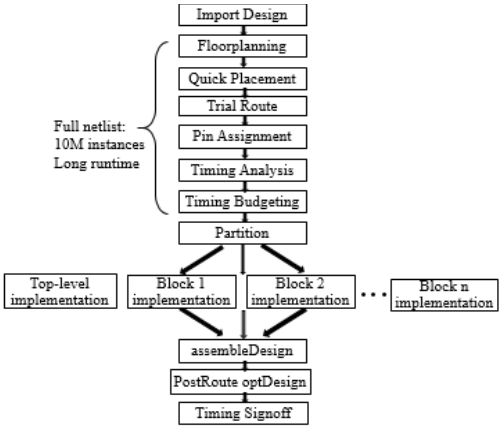

Fig. 1. Traditional hierarchical design

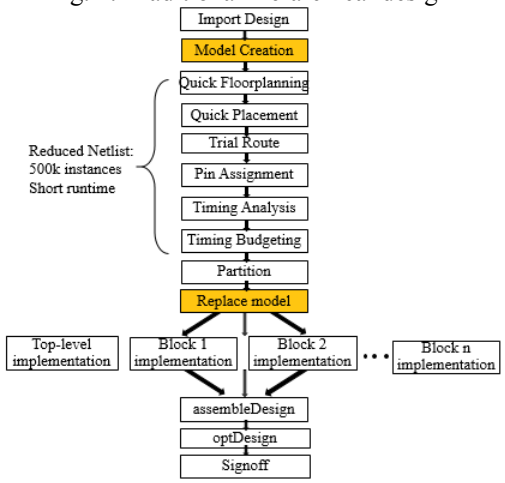

Figure 2. Hierarchical design of proposed method

To solve the problem of the increasing VLSI design complexity, hierarchical design method is widely used [8]. In this paper, the proposed method is to apply simplified model with hierarchical design method together to speed up floorplanning and the whole flow. The traditional hierarchical design process is shown in Figure 1, a 10M design need 10h for quick placement, 3h for trial route, 3h for timing analysis, 10h for timing budget, and then partition, block implementation and top implementation, assemble and optimization before timing signoff, finally the runtime of this method was more than 26h.

If adopting proposed method in hierarchical design as shown in Figure 2, although model creation took 5h, it could dramatically save runtime of the next series of processes, for example the quick placement took 2h and timing budget took 1h, so the whole flow runtime could be significantly reduced with the proposed method.

2.2 Active-logic reduction technology (ART)

Active-logic reduction technology (ART) ,use the concept of Logical Reduction in computer science, refers to remain the related logical units when automation calculation, it will mask the part which is irrelevant to calculation and generate an active logical view for computer to only see the active logic units in this view. In fact, it is just an active logical view that all original logic units remain in design actually. [9] Computer implements design optimization in the active logical view, and then maps the changed parts to the original view, but not changing inactive logic units which is irrelative with calculation. So in the way can reduce invalid calculation to improve calculation efficiency and reduce runtime [10]. The principle of ART technology in this paper is shown in Figure 3. The left graph shows an entire module in traditional hierarchical design whose logics in the partition are not visible, so it only can get boundary port definition and information. We know design optimization is based on timing prediction, such as output delay, but the timing prediction is completed before partition, its values will deviate from actual timing values. So it will take plenty of time for multiple iterations to achieve time closure through correcting the deviation between partitioning and assembling. The right figure uses ART technology, which can show the visible interface logic part in partition boundary, also can distinguish the interface logical path, the internal logic and loop path according to their timing information. Then use ART technology to mask inactive logic paths that are irrelevant to computation, and remain active logic interface logics that are relevant to computation. In this method, the timing prediction is closer to reality, so it can reduce the runtime of iterations optimization. This process is to create simplified model by reducing internal inactive logical units and connectivity in active logical view without changing original inactive logic units, but not affect floorplanning's quality while improving optimization efficiency and speed up the runtime, and can replace model with original design after partition.

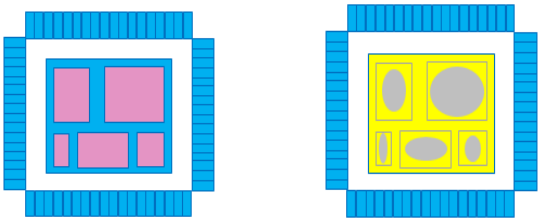

Figure 3. Diagram of ART technology

2.3 Simplified model and flex filler

Logic minimization algorithms has been used in VLSI synthesis, it is an important step forward in automatic logic synthesis [10]. Quick floorplanning method in hierarchical physical design flow can effectively reduce design size by reducing irrelevant logics, so as to quickly predict design implementation. The core of this method is simplified model, which is an approximate model based on ART technology to replace full netlist with a generating reduced netlist. Due to floorplanning pay more attention to interface connections rather than internal logics, simplified model aims to remove internal irrelevant units of modules. Then replace it with filling units named flex filler and generate a new reduced netlist file that is more conducive to quick floorplanning. As shown in Figure 4, the simplified model uses ART technology to replace the internal irrelevant logical units with flex fillers in netlist and only remain some logics between hard macros and modules. Therefore, the reduced netlist only contains related interface logics, it not only reduce information in netlist to speed up design but also not weaken floorplanning's quality. Generally to reduce about 90% of design size is the best.

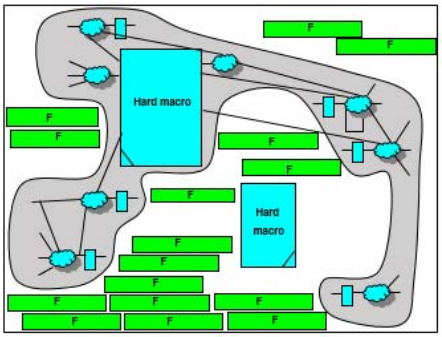

Figure 4. Schematic diagram of simplified model

Simplified model removes internal inactive logics, mainly is reg2reg logical units. Flex fillers as alternative

units are the special units which simulate the location and area of removed logical units to remain original utilization and area for more accurate floorplanning. In addition, there are other advantages of flex filler: Firstly, a flex filler's area is dozens larger than standard modules, so it can relatively reduce the number of modules in netlist to reduce design scale by simplifying internal logics. Secondly, flex filler itself does not have any logical connection, so P&R do not have to consider their connections which can save runtime. Also, floorplanning is supposed to adjust by results of P&R, so the reduction will speed up timing analysis and evaluation of floorplanning while reducing P&R runtime, finally overall floorplanning runtime can be reduced to a large extent. Figure 5 shows the flex fillers in EDA tool's physical view, selected rectangular box is flex filler whose size is much larger than the maintained interface logics in lower left corner of Figure 5.

Figure 5. Flex filler of physical view in EDA tool

3. Experimental results and analysis

This paper was implemented on Cadence Inc's EDA physical design software "Innovus"(the original name is Encounter), experimented on a 2.1M automotive electronic chip design and 5 other chips.[11] Figure 6 is the chip without any physical design. The simplified model based on ART technology is a physical level simulation model with reduced netlist, replace inactive logics with flex fillers and remain active logics.

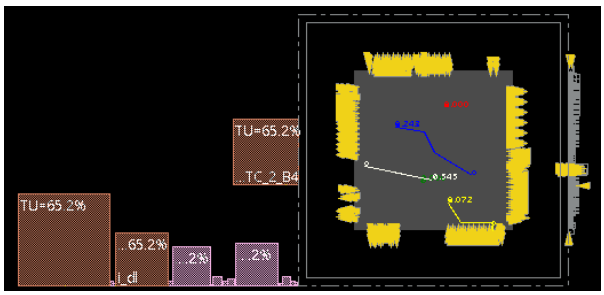

Figure 6. Initial chip without physical design

The experiment defined 89 modules as simplified models and gave a detailed model creation report in Figure 7. The initial instances is 2086769, about 2.1M, after modeling, it declined to 128695+71829=0.2M, so this method reduced design size from 2.1M to 0.2M, decreasing to 9.6% of original netlist. With reduced netlist, quick floorplanning method was easier to implement timing analysis, quick placement, preroute, optimization and budgeting. To prove it, we recorded the runtime of each step before partition step in hierarchical design flow and compared the runtime between design flow without model and with model. As shown in Table 1, without simplified model, total flow runtime before partition step was 807.95 minutes, but with simplified model, total flow runtime before partition step was 145.08 minutes. So runtime reduced by 5.57 times, and peak memory decreased by 2.22 times from 20G to 9G. Besides, in Figure 8, it is the chip floorplanning result with simplified model, and the simplified models was marked with coloured squares.

Actually, the method not only can dramatically speed up hierarchical physical design flow, it would also keep good timing and congestion results on one-pass flow. In Table 2, it used RC extraction engine to report timing after assembling design, there were all 6.87e+05 paths, and we focused on worst negative slack (WNS) of "reg2reg" -0.048ns, which was reasonable for one-pass flow when we evaluated design's time closure. The congestion reports showed the overall horizontal congestion was 0.10% and vertical congestion was 0.35%, which was acceptable in a 2.1M design.

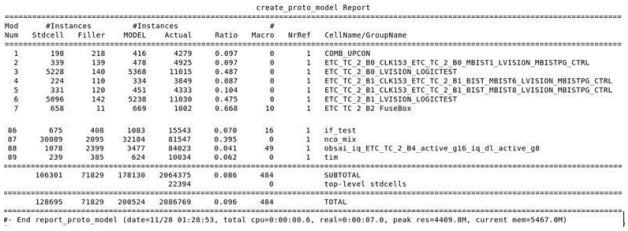

Figure 7. Report of creating simplified model

Table 1. Runtime comparisons of each step before partition step in hierarchical physical design flow

Approach	Without model	With model	Runtime improvement
Num of instance	2.1M	172K	N/A
Model Creation	N/A	14.78min	N/A
placeDesign	169.42min	21.70min	7.81
trialRoute	22.95min	2.47min	9.29
pinAssignment	0.52min	0.23min	2.26
OptDesign	542.30min	63.35min	8.56
Time Budgeting	31.58min	1.82min	17.35
Other steps(including save/restore design)	41.18min	40.73min	1.01

Total flow runtime before partition step	807.95min	145.08min	5.57
Peak memory	20G	9G	2.22

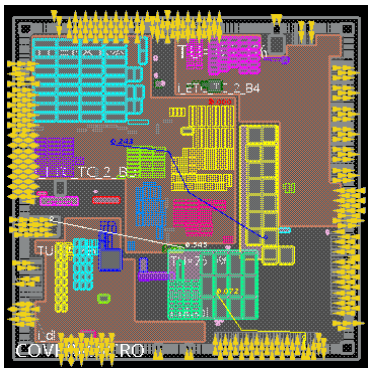

Figure 8. Chip design result of floorplanning.with simplified model

Table 2. Timing report after assembling design

Setup mode	all	reg2reg	default
WNS(ns)	-0.050	-0.048	-0.050
TNS(ns)	-2.912	-1.300	-1.612
Violating path	311	149	162
All paths	6.87e+05	6.86e+05	432

In order to further verify the effectiveness of proposed method, we experimented on several different huge size of designs to compare traditional method and proposed method in hierarchical physical design flow. In Table 3, the quick floorplanning method improve runtime by 6.2 times and reduce memory by 2.8 times on average. It provides the foundation for smooth implementation of P&R in hierarchical physical design and enhance the performance of EDA tool in VLSI physical design.

Table 3. Different designs using the proposed and traditional method flow comparison

No.	Design size	Runtime improvement	Memory reduction
1	2.1M	5.6	2.2
2	2.2M	3.9	2.0
3	2.6M	5.2	2.6
4	2.8M	5.4	2.3
5	3.5M	7.2	3.2
6	8.8M	9.7	4.3
average		6.2	2.8

4. Conclusion

With the rapid increase in size and complexity of VLSI, EDA tools are used in more effective way to meet speed and quality requirement of IC chip design. Floorplanning is the basis for P&R in physical design, but it usually accounts for about one-third runtime in whole flow to find a suitable floorplan, so shortening the floorplanning and evaluation runtime is an effective way to speed up chip's physical design. Using the logical reduction method to create model for floorplanning in top-down hierarchical physical design flow can effectively reduce design size by reducing irrelevant information of this step in netlist, then it can quickly predict if design meets timing and congestion requirement after physical design implementation. It proved the method can drastically reduce the runtime by 6.2 times and memory by 2.8 times on average in VLSI hierarchical physical designs according to the six comparison experiments.

References (follow the examples please)

[1] S.N. Adya and I.L. Markov, "Fixed-outline Floorplanning: Enabling Hierarchical Design", *IEEE Trans. on VLSI 11(6)*, pp. 1120-1135, 2003.

[2] Zhang K."Challenges and opportunities in nano-scale VLSI design", *IEEE VlSI-TSA International Symposium on VlSI Design, Automation and Test*, pp.6-7, 2005.

[3] R. Otten. "Automatic floorplan design," *19th Design Automation Conference*, pp.261-267, 1982.

[4] D.F. Wong, C. Liu. "A new algorithm for floorplan design," *23th Design Automation Conference,* pp. 101-107 1986.

[5] Li L, Ma Y, Xu N, et al. "Floorplan and Power/Ground network co-design using guided incremental floorplanning" *IEEE, International Conference on ASIC*, pp.747-750,2009.

[6] C.-W.Liu and Y.-W.Chang, "Floorplan and power/ground network cosynthesis for fast design convergence," *in Proc. of ISPD*, pp.86, 2006

[7] Chang B, Jigang W, Srikanthan T, et al. "Fast evaluation-based algorithm for fixed-outline floorplanning", *Computer Engineering and Technology (ICCET), 2010 2nd International Conference on. IEEE,* pp. V2-81-V2-85, 2010.

[8] Dai W J, "Hierarchical physical design methodology for multi-million gate chips", *International Symposium on Physical Design. ACM*, pp. 179-181, 2001.

[9] Reiner D, Tan M, Ventikos P, et al. "System and method for logical view analysis and visualization of user behavior in a distributed computer network": US, US 7165105 B2 [P], 2007.

[10] Brayton R K, Mcmullen C T, Hachtel G D. "Logic Minimization Algorithms for VLSI Synthesis" Kluwer Academic Publishers, pp. 537-543, 1984.

[11] Cadence. Innovus(Encounter) User Guide, Product Version 16.2, 2016