Precision Methodology: Transforming VLSI Physical Design with Tailored Floorplan and Power Plan Strategies

1st Anirudha Behera
MSc student, dept. of ECE
Illinois Institute of Technology
Chicago, USA
abehera1@hawk.iit.edu

2nd Dr. Ken Choi
Professor, dept. of ECE
Illinois Institute of Technology
Chicago, USA
kchoi12@iit.edu

***Abstract*—Navigating the complexities of the modern VLSI industry poses significant challenges in achieving the desired Quality of Results (QOR) for IC designs. The traditional methods, shaped over decades, are evolving to meet the dynamic industry needs. In response to the ever-changing landscape post-Moore era, this paper proposes an innovative and tailored approach to address a pivotal stage in VLSI physical design flow—the Floorplan. Emphasizing the critical role of efficient floorplanning in subsequent stages like Placement, CTS, and Routing, our approach aims to redefine and optimize the chip design process. This method, applicable to both top-down and bottom-up physical design flows and across node arenas, showcases how a specifically tailored strategy can lead to the creation of highly efficient, error-free floorplans and power plans. By adopting this approach, designers can pave the way for a more robust and productive IC design process, addressing the evolving challenges of the VLSI industry.**

***Index Terms*—Very Large Scale Integration (VLSI), Floorplan, Power plan, Quality of Results (QOR), Place and Route (PNR), Clock Tree Synthesis (CTS), Routing, Static Timing Analysis (STA), Physical verification (PV), Power and Ground Grids (PG Grids), Electronic Design Automation (EDA)**

I. Introduction

The landscape of Very Large Scale Integration (VLSI) design is currently undergoing a trans-formative shift, characterized by the intricate interplay of evolving technologies and escalating industry demands. Physical Design, considered the backbone of achieving high-performance chip design, plays a pivotal role in the overall design flow. Often referred to as the back-end design flow, it commences from the Synthesized Gate Level netlist and extends to the error-free GDSII file [1]. The subsequent intermediate phases in this process are depicted in the below Fig. 1, forming the standard flow of a chip after the design of RTL code and the execution of Logic Synthesis in what is known as the front-end design flow. While new advanced Electronic Design Automation (EDA) tools have undoubtedly eased the Physical Design flow, they are not entirely autonomous. In the floorplanning stage, the complex netlist typically comprises tens of millions of gate-level units. In 1982, R. Otten proposed an automatic floorplan design method Completing floorplanning and place-and-route (P&R)

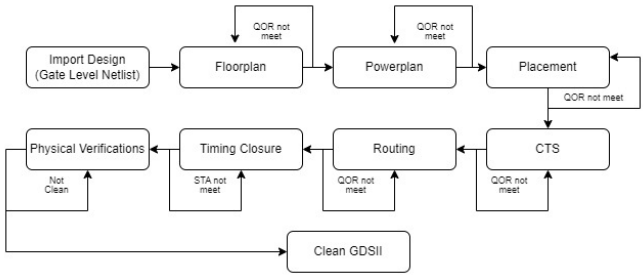

Fig. 1. Physical Design Flow

tasks quickly and accurately poses challenges for Electronic Design Automation (EDA) tools [4] . Consequently, EDA tools necessitate the incorporation of several constraints to facilitate efficient floorplanning. In floorplanning, the main goal is to optimize a predefined cost function or cost metric, which can be only the area given by $f(F) = W(F) \times H(F)$, where F is a floorplan [6].

Within the back-end flow, the Floorplan and Power Plan emerge as the initial and most crucial phases, providing designers with control over up to 80% of the Power, Performance, and Area metrics [9]. A meticulously planned floorplan has the potential to yield favorable timing and congestion outcomes by reducing the chip area to minimize the chip cost [7] [8]. The sooner we can iterate and enhance the floorplan, the more expeditiously the design is likely to reach its PPA objectives. Nevertheless, determining the precise percentage of PPA control attainable in these stages is contingent upon factors such as design specifications, optimization methodologies, and tool efficacy.

Consequently, providing an unequivocal answer becomes challenging, and instead, we can offer a general estimate grounded in assumptions and established best practices.

A potential method for approximating the percentage of Power, Performance, and Area (PPA) control during the floorplan and power plan stages involves comparing the PPA metrics before and after these phases. By assessing the initial

PPA metrics (P1, F1, A1 for power, frequency, and area) and the final metrics post floorplan and power plan (P2, F2, A2), the percentage of PPA control can be calculated using the formula:

$$PPA\ Control = \frac{(P1 - P2) + (F2 - F1) + (A1 - A2)}{(P2 + F2 + A2)} \times 100\%$$

This formula estimates desirability for lower power, smaller area, and higher frequency, using normalized PPA metrics. Caution is advised, as it may not consider trade-offs or critical factors like reliability, yield, and cost.

II. Approach

To achieve maximum PPA optimization during Floorplan and Power Plan stages, it is crucial to adopt an effective and error-free approach. This ensures obtaining results without encountering significant errors in complex high-performance designs. In this context, our approach stands out as unique and result-oriented compared to the traditional methods of Floorplan and Power plan.

A. Traditional Approach of Floorplan and Power plan

Traditionally, the Floorplan and Powerplan stages have been treated as separate entities in the chip design process. In the Floorplan stage, the shape of the block may either be plotted (Bottom-up) based on design considerations or provided (Top-down) by the Top-level design team as part of a Physical-Aware design. Die shapes typically fall into four categories: Rectangle, U, I, and L [2]. If the designer does not specify the shape, the default is a rectangle. The subsequent steps are outlined in the Pseudo code below.

1) **Sanity Check**
   - `check_netlist()`
   - `check_timing()`
   - `report_design_mismatch()`
2) **Initialize Floorplan**
   - `initialize_floorplan()`
3) **Set Block Pin Constraints for I/O Ports**
   - `set_block_pin_constraints()`
4) **Fix Physical Status of I/O Pins**
   - `fix_physical_status_io_pins()`
5) **Place and Fix Physical Status of Macros**
   - `fix_physical_status_macros()`
6) **Create Keepout margin for macros**
   - `create_keepout_margin()`
7) **Set and Compile Boundary Cell**
   - `set_and_compile_boundary_cell()`
8) **Create Tap Cells**
   - `create_tap_cells()`

In the Powerplan stage, the design involves creating power and ground grids for Macros, Standard cells, and other cells. The traditional approach to plotting the power and ground grid flow is detailed in the Pseudo code below.

1) **Create the PG Net**
   - `create_nets()`
2) **Making Logical Connection**
   - `connect_pg_nets()`
3) **Setting Attributes for TIE Cells**
   - `set_attribute()`
4) **Creating PG Rails**
   - `create_pg_pattern()`
   - `set_pg_strategy()`
   - `compile_pg()`
5) **Create Macro Pin Connection**
   - `create_pg_macro_conn_pattern()`
   - `create_pg_strategy()`
   - `compile_pg()`
6) **Create Standard Cell Pin Connection**
   - `create_pg_std_cell_conn_pattern()`
   - `create_pg_strategy()`
   - `compile_pg()`
7) **Creation of Vias between Rails and PG Straps**
   - `create_pg_vias()`

B. Proposed Approach of Floorplan and Power plan

The traditional approach, widely adopted by the industry for decades, is susceptible to errors, a topic we will delve into with comprehensive experimental results.

This realization has led us to introduce a novel approach that is highly reliable, swift, and error-free across various nodes. The proposed method advocates combining the design of Floorplan and Powerplan into a single stage by reorganizing key phases, resulting in a more resilient flow. Below is the refined Pseudo code for the combined Floorplan and Powerplan stage.

1) **Sanity Check**
   - `check_netlist()`
   - `check_timing()`
   - `report_design_mismatch()`
2) **Initialize Floorplan**
   - `initialize_floorplan()`
3) **Set Block Pin Constraints for I/O Ports**
   - `set_block_pin_constraints()`
4) **Fix Physical Status of I/O Pins**
   - `fix_physical_status_io_pins()`
5) **Place Macros**
   - `place_macros()`
6) **Create the PG Net**
   - `create_net()`
7) **Making Logical Connection**
   - `connect_pg_net()`

8) **Setting Attributes for TIE Cells**
   - `set_attribute()`
9) **Creating PG Grids**
   - `create_pg_pattern()`
   - `set_pg_strategy()`
   - `compile_pg()`
10) **Set and Compile Boundary Cell**
   - `set_and_compile_boundary_cell()`
11) **Create Tap Cells**
   - `create_tap_cells()`
12) **Create Standard Cell Rails**
   - `create_pg_std_cell_conn_pattern()`
   - `create_pg_strategy()`
   - `compile_pg()`
13) **Creation of Vias between Rails and PG Straps**
   - `create_pg_vias()`
14) **Create Macro Pin Connection**
   - `create_pg_macro_conn_pattern()`
   - `create_pg_strategy()`
   - `compile_pg()`
15) **Create Keepout margin for macros**
   - `create_keepout_margin()`
16) **Fix Physical Status of Macros**
   - `fix_physical_status_macros()`

In practical design scenarios, the specific execution of each step in the given pseudo code may vary, contingent upon the nature of cells employed and the unique objectives of the design. However, the overall sequential execution, as outlined in the pseudo code, remains consistent. The provided pseudo code corresponds to the design employed in our experimental studies for this specific topic.

III. EXPERIMENTAL RESULTS

This experiment was conducted on OpenSPARC T1 processor, utilizing the Synopsys EDA toolset with a focus on 14nm and 28nm technology nodes. The deliberate choice of testing on both 14nm and 28nm nodes aims to demonstrate the applicability of the proposed approach across a diverse node arena and to present the Quality of Results (QOR) of Power, Performance, and Area (PPA) in a comparative manner.

Both designs adhered to the standard Physical Design flow, commencing from Logic Synthesis, traversing through various intermediate stages, and culminating in Physical Verifications leading to GDSII tape-out. The opted results will be compared to the proposed Approach.

a) Initialize floorplan: At the outset of the Floorplan stage, precision and clarity in design aspects are imperative. The level of detail incorporated at this phase significantly influences the robustness of the ensuing design.

- `Shape`: The shape determination hinges on the size, orientation of macros, and overall design specifications.
- `Core Area Utilization`: Maximizing core area utilization is advantageous, striking a balance between efficiency and congestion avoidance.
- `Offset`: Preserving the distance between Core and Die boundaries is critical, particularly as I/O pins will be positioned in this region. Maintaining an optimal core area ensures minimal congestion between pins, reduces Die-to-Die variation, and contributes to a healthy packaging.

Node	Shape	Core Area Utilization	Offset
14nm	Rect	58%	4
28nm	Rect	72%	4

TABLE I
INITIALIZE FLOORPLAN

b) Place Macros: The placement of macros poses significant challenges in VLSI design, impacting congestion, routing, and rule violations. A flawed floorplan can lead to design failures. However, employing strategic tactics inspired by the discipline of Saturn, we can optimize the placement process and achieve superior results.

- `Hierarchy Exploration`: Utilize the "Hierarchy Exploration" feature in EDA tools to assess the design hierarchy of macro blocks. Place macro blocks with the same hierarchy together, aligning them with the nearest connected I/O pins, as depicted in Fig. 2. This technique enhances the efficiency of EDA tools and facilitates quicker timing analysis [5].
- `Orientation for Efficient Placement`: Maintain the orientation of macros facing towards the core region, where standard cells will be placed. Exercise caution to ensure that macros do not obstruct I/O pins, thus preventing connectivity issues.

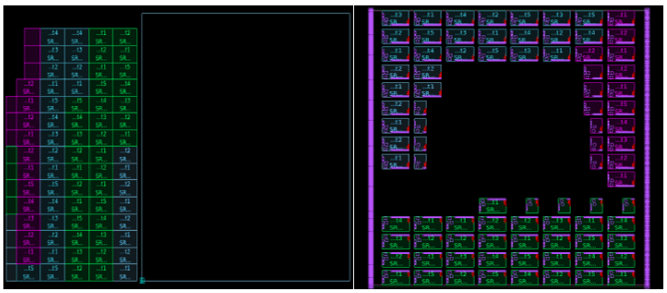

Fig. 2. Place Macro

c) Creation of PG Grids: Power (VSS) and Ground (VDD) grids play a critical role in providing power and ground connections to macros within a block [3]. Conventionally, PG Grids are established using the Pattern, Strategy, and Compile methods. However, for precise control over each grid, a direct approach is often employed.

The `create_pg_strap` command is instrumental in generating independent grid lines, where the following parameters are specified:

- `layer`: Denotes the layer name, typically higher layers are preferred.
- `direction`: Specifies the orientation, either horizontal or vertical based on design requirements.
- `width`: Signifies the width of the grid lines, a critical factor for achieving optimal Electromigration results.
- `net`: Indicates the net type, such as VDD or VSS.
- `start & stop`: Specifies the start and stop points of the grid lines.
- `pitch`: Represents the distance between similar layers in a particular direction.

To prevent potential errors in the future, meticulous calculation of start, stop, and pitch is imperative for each of the four individual layers, based on the orientation and spacing between placed macros. Alternatively, an approach involves initially plotting grids with average considerations and subsequently adjusting macros to align with the PG grid. This ensures that each macro pin can be efficiently connected to its nearest available VDD or VSS grid.

This strategic approach not only facilitates precise control over grid lines but also optimizes the power and ground distribution, contributing to enhanced Electromigration performance and overall robustness in VLSI design.

Fig. 3 provides a visual representation of the PG grid setup.

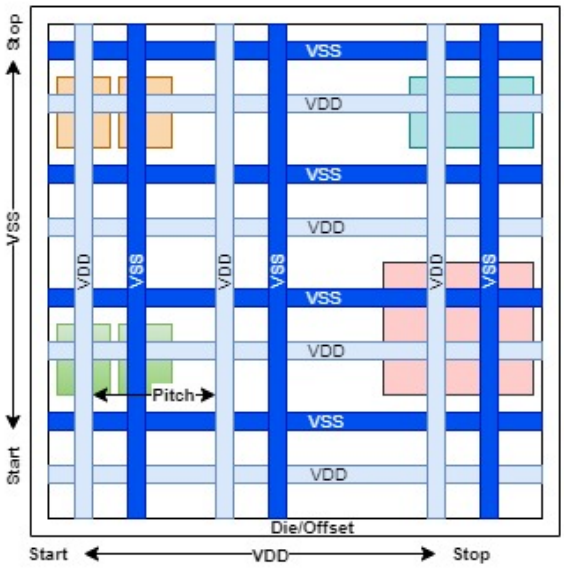

Fig. 3. Power and Ground Grid Setup

d) Creating Boundary and Tap Cells:: The creation of Boundary cells is a straightforward process involving the use of patterns, strategies, and compile methods to plot them along the border of the block's core area.

However, the placement of Tap cells introduces a level of complexity. Precise determination of the distance between each vertical rail of cells is crucial, particularly in areas where macros are situated. The goal is to maintain approximately equal distances on both sides of macros, ensuring consistency in the spacing between vertical VDD and VSS PG grids. Failure to achieve uniform spacing may result in floating errors in later stages, potentially leading to `check_pg_drc` errors.

To prevent `check_pg_drc` errors, it is essential to ensure that every single standard cell PG rail between macros and across Tap cells intersects the opposite net type of PG grids. This approach is illustrated in the Fig. 4 below.

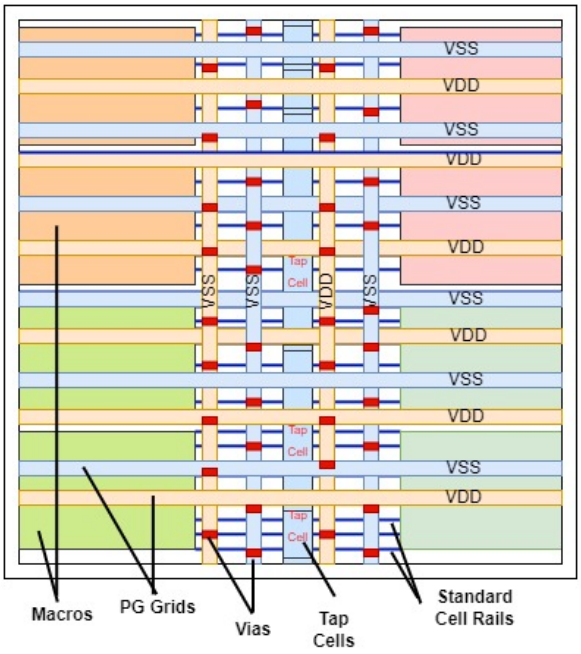

Fig. 4. Illustration of Tap Cell Placement Strategy

e) Create Standard Cell Rails: Standard cell rails are horizontally plotted VDD and VSS connections, typically implemented using lower-level, thinner metal layers compared to PG (Power/Ground) grids. These rails facilitate the interconnection of standard cells, boundary cells, tap cells, and other cells across the core area.

- `create_pg_std_cell_pattern`: The basic layer parameters are declared, including pattern type, layer name, and rules check.
- `set_pg_strategy`: In this step, the rail strategy for the core pattern is declared, specifying names and net types.
- `compile_pg_strategy`: The declared strategy is compiled in this step.

Once the standard cell rails are plotted, adjustments may be necessary for macros to prevent any short circuits between macro pins and rails. The above Fig. 4 illustrates the purpose of this section:

This process ensures the effective integration of standard cell rails and minimizes the risk of shorts between macro pins

and rails.

f) Creation of Vias between Rails and PG Straps: Vias are essential at the cross-sections where Power/Ground (PG) grids and rails of the same type intersect, creating connections between different metal layers. The creation of via stacks between lower and higher metal layers is crucial to prevent shorts.

- `create_pg_vias`: This command is utilized to create vias across the design where they are missing.
- `nets`: Specifies the net types for which via stacks should be applicable, such as VDD and VSS.
- `from_layer` to `to_layer`: Declares the lower-level layer from which the via stack will start and the upper-level layer it will cover.

This process ensures the insertion of vias at critical intersections, enhancing the connectivity between PG grids and rails and preventing potential shorts. In Fig. 4, vias have been depicted between the same net types and between PG grids and rails.

g) Creation of Macro Pin Connection: Macro pin connection involves the use of higher-level metal layers, such as PG grids, to connect the VDD and VSS of PG grids to each macro block. Implementing macro pin connection at the end of the design process serves a specific purpose, effectively reducing the number of `check_pg_drc` violations, especially in complex designs where these violations can be substantial.

The significance behind this approach lies in its ability to prevent unnecessary errors. When Macro pin connection is implemented first and then PG vias are created, the second command attempts to generate vias between macro pins and passing rails. This is unnecessary and can lead to errors, particularly when macro pins are situated nearby vertical PG grids. The following figure illustrates this scenario and its solution:

- `create_macro_pg_conn`: This command is used to create macro pin connections using a specified pattern, strategy, and compile method.

The depicted Fig. 5 illustrates a real-world scenario of a DRC (Design Rule Check) violation in the VDD Macro pin via of the OpenSPARC T1 28nm model. This violation occurs due to an unnecessary via on the Macro pin caused by the post-Macro pin connection via plotting. The left side of the figure showcases the error, emphasizing the importance of plotting vias before Macro pin connection to avoid such issues. In contrast, the right side figure demonstrates a scenario with no errors, highlighting the effectiveness of plotting vias before Macro pin connection.

IV. Conclusion

The innovatively proposed tailored integrated approach for Floorplan and Powerplan design marks a significant departure from conventional methodologies, introducing a reliable, expeditious, and error-resistant process. Experimental validation conducted on the OpenSPARC T1 processor across both 14nm and 28nm nodes underscores the practical applicability and efficacy of the approach, showcasing commendable Quality of Results (QOR) for Power, Performance, and Area (PPA). Strategic macro placement, meticulous PG grid creation, and precise via handling emerge as tailored strategies, enriching the robustness of the design. This pioneering approach represents a promising leap forward in VLSI design practices, promising heightened efficiency and a diminished likelihood of errors in the intricate process of Floorplan and Powerplan synthesis. The tailored strategy proposed in this paper, while simple, proves remarkably effective when executed with the right methodology, as outlined.

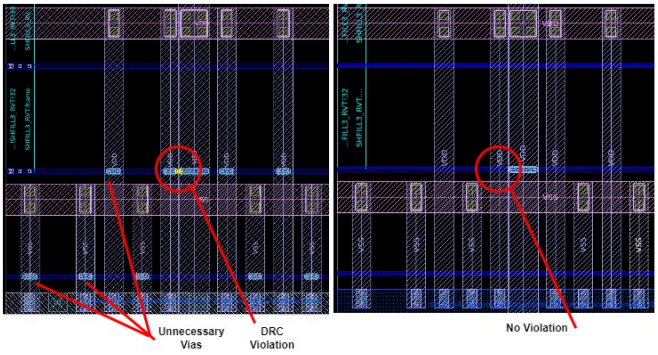

Fig. 5. Illustration of proposed strategy for Macro pin placement

Node	Power	Performance	Area
14nm	5.25 pW	250 MHz	168642
28nm	7.01 pW	250 MHz	985543

TABLE II
PPA METRICS OF FINAL GDSII

References

[1] G. P. J, S. R. Karbari, S. Ammikkallingal, and S. K. Bellal, "Analysis, Physical Design, and Power Optimization of Design Block at Lower Technology Node," in *2018 3rd IEEE International Conference on Recent Trends in EICT (RTEICT-2018)*, May 18th & 19th, 2018.

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

[3] 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.

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

[5] Y. Zhou, Y. Yan, and W. Yan, "A Method to Speed up VLSI Hierarchical Physical Design in Floorplanning," in *2017 IEEE Conference Name*, 2017, pp. 347-350.

[6] Guolong Chen, Wenzhong Guo, Hongju Cheng, Xiang Fen, and Xiaotong Fang, "VLSI Floorplanning Based on Particle Swarm Optimization," Proceedings of 2008 3rd International Conference on Intelligent System and Knowledge Engineering, 2008.

[7] B.Sowmya, Sunil M.P., "Minimization of Floorplanning Area and Wire Length Interconnection Using Particle Swarm Optimization," International Journal of Emerging Technology and Advanced Engineering, vol. 3, Issue 8, August 2013.

[8] Tsung-Ying Sun, Sheng-Ta Hsieh, Hsiang-Min Wang,and Cheng-Wei Lin, "Floorplanning Based on Particle Swarm Optimization," Proceedings of the 2006 Emerging VLSI Technologies and Architectures (ISVLSI'06), 2006.

[9] Jarmo, Physical Design (PD) – Floorplanning, LMRSuomi, https://lmr.fi/int/physical-design-pd-interview-questions-floorplanning