Logical-Physical Co-design for Deep Submicron Circuits: Challenges and Solutions*

Massoud Pedram
Department of Electrical Engineering – Systems
University of Southern California
Los Angeles, CA 90089-2562
Tel: 213 740 4458
Fax: 213 740 7290
E-mail: massoud@zugros.usc.edu

Abstract – As IC fabrication capabilities extend down to sub-half-micron, the significance of interconnect delay and power dissipation can no longer be ignored. Existing enhancements to synthesis and physical design tools have not been able to solve the problem. The only remaining alternative is that tradeoffs in logical and physical domains must be addressed in an integrated manner. Vast business opportunities will be lost unless more revolutionary changes to design flow are made. This paper discusses three technologies which are key to performing logic synthesis and physical layout optimization in tandem. They are early floorplanning, layout-driven logic synthesis, and post-layout resynthesis.

I. Introduction

Today's ASIC design teams face a number of difficult challenges. Prominent among them are shortening design cycles resulting from increasing time to market (or time to volume) and the growing complexity of designing at 0.25 μ and 0.18 μ.

During the process of designing high performance VLSI circuits, designers often find that their designs do not meet the timing and/or power constraints after layout. This is a situation that mainly arises due to the weak interaction between logic synthesis and physical layout optimization tools. Synthesis which has the ability to significantly alter the timing and power dissipation of the circuit, uses a relatively simple model of wire loads. By comparison, physical design which has accurate wire loads from back-end extraction tools, does not alter the gate implementation and hence cannot drastically change the timing/power profile of the circuit.

This problem is compounded as feature sizes shrink to quarter-micron and below i.e., the so-called deep submicron (DSM) process technologies. ASIC designers enter a new era characterized by the following:

- Device count per chip is increasing rapidly (Moore's law states that it doubles every 18 months). The latest processes, with 0.25 μ features, can achieve up to 40,000 gates per square millimeter, compared with fewer than 10,000 gates/mm^2 for a 0.35 μ process. At the same time, the die dimensions are increasing beyond 25 mm on a side.
- Gate count per ASIC design is increasing at an even faster rate due to the incorporation of internal and third-part cores (IPs). Indeed, although the average design start for ASICs was estimated to be about 100 K-gates in 1996, the number of large designs (system-level integration on a chip) is growing rapidly.
- As the width of wires shrinks, resistance increases more rapidly than the capacitance decreases. Consequently, in DSM design, the effects of interconnect delay dominate the chip timing. For 0.5 μ technologies, interconnect delays can account for more than 50% of total delay on a typical net. For 0.35 μ it can account for more than 70%.
- Mismatches between predicted delays after synthesis and actual delays after layout can be as much as 100-200%, causing a significant increase in design iterations and turnaround time to first silicon.
- Manufacturing process variations, which appear as

*This work is supported in part by a Presidential Early Career Award for Scientists and Engineers.

spatial variations in parameters such as L, V_t and interconnect capacitance, can greatly impact the chip timing.

- Complicated second-order effects such as edge rate effects, signal coupling and ground bounce, and process and temperature variations greatly impact the chip performance. In particular, a large height-to-width ratio - 2 to 1 in 1998 - and many interconnect layers make lateral coupling increasingly more significant than ground coupling. Hence, delay cannot be calculated accurately without taking crosstalk into account.
- Number of nets that are at a performance risk increases greatly. It is expected that this number will be in tens of thousands for a large ASIC in a DSM process technology.
- The sheer complexity of 20-million-gate chips chokes design tools and causes exponentially increasing design iteration cycles. Today's design tools are outpaced by the capacity and performance requirements of the ASICs.

For more detailed description of the DSM process technologies and design trends, refer to the National Technology Roadmap for Semiconductors (NTRS) [1].

Design technologies must therefore be advanced for design flows, methodologies, tools, and standards to be able to produce chip designs with 100 million or more transistors with the same number of designers in the same time it takes now for a 5 million transistor chip. Without these advances, the semiconductor and electronics industries will suffer an economic death as they fall off the productivity curve (25-30% improvement in $ per function) that the world has come to expect year after year [2].

To cope with DSM design challenges, a different design methodology is required where the sharp division between logical and physical design disappears. What separates this methodology from the traditional flows is that synthesis is driven by physical design considerations and that it focuses on interconnections rather than gates as the major contributing factor to circuit delay and power dissipation.

In this new design paradigm, the front end optimizations are globally interleaved and locally integrated with the back end optimizations. The front end tools receive accurate interconnect parasitics data and power/delay/signal integrity estimates from the back end tools, and pass detailed logical information and power/timing constraints to the back end tools. For example, the RT-level description of a design can go through the following locally integrated optimization loops:

1. Logic partitioning (with replication), early floorplanning, inter-block global routing, and block-level delay budgeting
2. Logic synthesis (restructuring and mapping), loose placement, and gate-level slack assignment
3. Gate sizing, buffering and re-wiring, detailed placement, and timing recalculation
4. Wire sizing and spacing, pin swapping, detailed routing, and cross-talk analysis.

Notice that this is only one possible flow and many other flows are possible.

Three key technologies which are central to the success of any DSM design flow are listed below. The list does not include back-end tool requirements such as low level timing analysis and optimization for interconnects (see [3]), parasitic extraction and signal integrity verification (see [4]), or detailed routing tools here, although they are obviously critical to any flow.

1. Circuit partitioning and early floorplanning technologies that better integrate the estimation and analysis of logical, physical, timing, and power representations of a design and help manage the design complexity.
2. Layout-driven logic synthesis techniques which either perform logic synthesis concurrently with placement or attempt to identify and minimize an abstract measure of routing complexity during synthesis.
3. Post-layout resynthesis techniques which perform local netlist optimizations after detailed placement and routing.

These techniques are not mutually exclusive; that is, it is possible (and desirable) to employ all of them in the DSM design flow.

In the remainder of this paper, we will discuss each of these techniques. The conclusion of the paper compares these techniques.

I. Early Floorplanning

After a behavioral/RT-level description of a DSM design is created and verified, the ASIC designers must

get the circuit to meet the timing specifications. Because of the dominance of interconnect delays, they must floorplan their designs and extract timing delays of long nets to ensure that the design meets the timing requirements.

For large ASICs, design tools partition the circuit into more manageable blocks of 5,000 to 15,000 gates each as follows. First, the logical hierarchy of the design is analyzed to determine which portions of the hierarchy are well structured for problem-free physical implementation, and which portions are unstructured and must be processed further to avoid routability and/or timing problems. Physical partitioning and regrouping algorithms are then used on the unstructured logic to repartition the design for better physical implementation.

The ASIC vendors provide two-dimensional look-up tables which supply the average net capacitance as a function of the number of pins in the signal net and the size of the block where the net resides (assuming square shape for the block). This statistical wire load approximation was effective for process geometries above 0.8 μ. It however fails for the DSM process geometries. The approach that most ASIC designers are taking is to create a floorplan after synthesis. The floorplan can illuminate the timing violations and routing congestion. The floorplan can then be adjusted to minimize the timing, crosstalk, etc. problems and the adjusted floorplan is returned to the synthesis tool, which then attempts to eliminate the timing problems by in-place optimization (i.e., gate sizing and buffering). The problem with post-synthesis floorplanning is that the synthesizer does not know about the relative placement of logic blocks before it creates a gate-level description of each block. In contrast, pre-synthesis (early) floorplanning estimates block areas (based on its "understanding" of the synthesis operations or by using a "quick synthesis" tool), assigns shapes, positions and pin directions to the blocks, calculates the inter-block delays, and finally performs delay budgeting (slack assignment) for each block. This information then drives the synthesis of each block and often results in far fewer design iterations.

An early floorplanner provides behavioral/RT-level topology and analysis for optimizing the physical design early in the design cycle. The result is greater predictability and control over the physical design for the logic designer, and greater efficiency and effectiveness for the physical designer during physical layout. By analyzing the consequences of physical layout early in the process, designers reduce the risk of post-layout iterations. An RT-level design planner is nothing but an IC floor-planner (e.g., [5] and [6]) with RT-level delay, signal integrity, and power estimation and budgeting tools. With the added information, the designer can make more detailed decisions about logical and physical partitioning, module architectures, clocking and power structures, and the routing topologies of critical nets. Examples of commercial RT-level floorplanners are Preview[tm] from Cadence and Planet-PL[tm] from Avant!.

I. Layout-Driven Logic Synthesis

Two basic approaches for performing wire load optimization during logic synthesis have been proposed. They are placement-based and structure-based approaches. In the placement based approach, optimization is guided by information derived from a "companion placement" solution for the circuit being synthesized. In the structure based approach, optimization is performed by using an abstract cost measure which captures the routing overhead/complexity of the circuit.

Placement-based approach

During synthesis, the wire loads are unknown, and are traditionally modeled using the statistical wire load estimates as explained previously. These statistical estimates are, however, failing to accurately predict the circuit delay in DSM designs. This is because the variance of the actual wire load versus pin-count and block-size plots is increasing in DSM designs, thus, the wire load estimator exhibits a significant error on a net by net basis. It can thus be concluded that to determine the circuit delay accurately, the cell positions must be known.

A concurrent placement and technology mapping algorithm was first proposed in [7]. The overall flow is as follows:

- A companion placement of the boolean network is generated.
- This placement information is used to guide the logic synthesis tool to optimize post-layout costs.
- As the network is modified during logic synthesis, the placement is dynamically updated.
- A combined synthesis and placement solution is eventually obtained.

The main advantage of this approach is that once the cell positions are known, it becomes possible to reliably estimate the routing overhead of every net in the circuit.

In [8], the authors proposed to use the gate positions derived from the companion placement solution to obtain a linear order on the fanouts of gates. Next a fanout optimization algorithm is developed that generates fanout trees which are free of internal edge crossings. This is achieved by using a special type of fanout trees called *alphabetic trees*. These fanout trees provide a good trade-off between circuit performance and routability. These are trees that minimize (maximize) arrival (required) time at the root of the decomposition (fanout) tree subject to a fixed linear order on the sinks, without creating any internal edge crossings which in turn results in lower routing overhead. The delay penalty for using alphabetic trees is small; it can be shown that under the unit delay model, increase in depth is at most one for optimal alphabetic fanout trees as compared to optimal non-alphabetic trees. In [9], the authors enhance FPGA routability by combining technology mapping with some amount of place-and-route. More precisely, they reduce wiring congestion in an FPGA device by moving "iotas" of logic between CLB's after the initial placement. Hierarchical approaches for concurrent FPGA mapping and placement are proposed in [10] and [11]. In [12], the authors propose a layout-driven synthesis approach for FPGAs which is based on identification of alternative wires and alternative functions for wires that cannot be routed due to limited routing resources in FPGAs. Their results demonstrate that routing flexibility can be significantly improved by considering the alternative wires.

The authors of [13] recently reported an optimal algorithm for solving the simultaneous technology mapping and linear placement problem for trees using dynamic programming. The resulting mapped and placed circuit is guaranteed to use the minimum post-layout area (including the routing area required to complete all connections within the tree). The basic idea is to combine dynamic programming approaches for tree-based technology mapping (such as [14]) with dynamic programming approaches for minimum cut-width linear placement of trees (such as [15]). The authors also describe a floorplan-driven simultaneous mapping and placement algorithm for general directed acyclic graphs (DAGs). The outline of the algorithm is as follows:

- Partition the initial DAG into a set of trees.
- Treat each tree as a soft macro-cell and floorplan the circuit.
- Perform global routing, timing calculation and budgeting.
- Do simultaneous technology mapping and linear placement for each tree using [13].

Preliminary results show that this integrated approach improves the circuit performance by 25% compared to the conventional flow which separates technology mapping from gate placement.

The authors of [16] reported POINT, a timing-driven placement with an integrated netlist optimization engine. The idea is to interleave global placement and (bi or quad) partitioning step of GORDIAN-like placement [17] with signal substitutions to change the circuit structure and improve its delay [18]. Accurate delay estimations for the wires are possible because of the companion placement solution. This technique can be extended by incorporating more powerful logic restructuring techniques; however, a key issue is to ensure that incorporation of these operations in the innermost loop of the placement program does not cause non-convergence of the overall procedure. Preliminary results show 18% reduction in circuit delay compared to the flow which performs netlist optimization and timing-driven placement separately.

Structure-based approach

The principal idea behind the structure based approach is to identify network parameters which affect the routing overhead in the circuit, and then derive easy-to-compute cost measures which correlate well with the actual post-layout cost. The advantage of this mechanism is that it attempts to correlate excessive routing requirement in a synthesized design to the structure of its underlying directed graph. These network parameters will therefore be abstract and hence, independent of the placement/routing tools, or the process technology used. Such abstract costs - including network cut-width at different logical depths, or fanin/fanout signal ranges or signal overlaps - may also be easier to compute and can be effectively used during earlier stages of logic synthesis. The primary difficulty associated with this approach is whether such abstract parameters exist, and if so, whether they can be used to derive a cost measure which will consistently generate circuits with improved post-layout area. One objective of this research is to answer these questions and derive such cost functions.

The first published work in this area is [19] where the authors propose the concept of lexicographic extraction. The idea is to incrementally construct and impose a partial order on the input signal variables of a two-level logic circuit as common subexpressions (kernels) are extracted. Future extractions must abide by the derived input variable order. The rationale is that by ordering the signals that merge to create the extracted kernels in the resulting multi-level logic cir-

cuit, the routing cost for the circuit is reduced. In [20], the authors used the fanout ranges of the nodes in the circuits to achieve better signal localization and to achieve uniform distribution of signals across the network. Experimental results, although non-conclusive, are promising.

I. Post-Layout Resynthesis

An effective technique for solving certain timing violations in the circuit is to use logic re-synthesis based on back-annotated parasitic capacitance and gate delay information obtained after placement and routing. In many cases, using only operations such as gate resizing, buffering, and small logic changes, the original placement and global routing solutions can be substantially preserved. Therefore, the iteration between logic re-synthesis and physical design converges quickly.

Evidence is emerging that these techniques can be successful in fixing the timing and load related violations which may remain after timing driven placement. For example, in [21], the authors start from an initial placement and perform fanout optimization and gate sizing to improve the circuit delay. Net delays are estimated directly based on the initial placement solution and current gate sizes and fanout tree structures. In [22], the authors presented a technology re-mapping and re-placement algorithm for alleviating routing congestion in a bit-sliced layout. Experimental results showed 20% improvement in circuit area. Such an improvement in routing density could not, however, be achieved using purely topological/physical operations (such as pin permutation, cell swapping, lateral shifting, etc). In [23], a discrete gate sizing algorithm is presented. Although the formulation is non-linear and non-convex, the heuristic solution obtained by the author shows the potentially large impact of gate sizing on circuit delay after layout is completed. An example of a post-layout transistor resizing tool is AMPS[tm] from Synopsys.

I. Conclusion

The real question is not whether logic synthesis and layout optimization will have to merge. The question rather is how this merge will take place. Logic synthesis and physical design systems are both complicated software systems, each having its own representation and performance models. The three types of techniques discussed in this paper show promise in closing the gap between logic synthesis and physical design. Among the three, early floorplanning is the most developed and commonly employed technique, followed by post-layout resynthesis, and layout-driven logic synthesis (which is just beginning to receive attention by the EDA community). Layout-driven synthesis however holds the biggest promise in addressing the DSM design challenges.

Other key problems include development of predictable, timing-driven synthesis and layout optimization algorithms, adoption of EDA standards and common databases to support the integration of layout and synthesis tools, and introduction of structured design styles that offer lower wiring overhead by construction.

References

[1] Semiconductor Industry Association, *National Technology Roadmap for Semiconductors,* 1994.

[2] Richard Bushroe, In "Panel: physical design and synthesis: merge or die," M. Pedram (moderator), *Proc. 34th Design Automation Conf.,* June 1997, pages 238-239.

[3] J. Cong, L. He, C-K. Koh and Z. Pan, "Interconnect design for DSM ICs," *Proc. Int'l Conf. on Computer Aided Design,* November 1997, pages 478-485.

[4] W-M. Dai and W. Sun, "3-D parasitic extraction for DSM IC design," *Integrated System Design,* July 1997, pages 22-30.

[5] T-C. Wang and D-F. Wong, "Optimal floorplan area optimization," *IEEE Trans. on Computer-Aided Design,* July 1992, pages 992-1002.

[6] M. Pedram and E. S. Kuh, "BEAR-FP: a robust framework for floorplanning," *Int'l Journal of High Speed Electronics and Systems,* Vol. 3, No. 1 (1992), pages 137-170.

[7] M. Pedram and N. Bhat, "Layout driven technology mapping," *Proc. 28th Design Automation Conf.,* June 1991, pages 99-105.

[8] H. Vaishnav and M. Pedram, "Routability driven fanout optimization," *Proc. 30th Design Automation Conf.,* June 1993, pages 642-647.

[9] N. Bhat and D. Hill, "Routable technology mapping for LUT FPGA's," *Proc. Int'l Conf. on Computer Design: VLSI in Computers and Processors,* October 1992, pages 95-98.

[10] C-S. Chen, Y-W. Tsay, T-T. Hwang, A-C. Hu, and Y-L. Lin, "Combining technology mapping and placement for delay optimization in FPGA designs," *Proc. Int'l Conf. on Computer Aided Design,* November 1993, pages 123-127.

[11] N. Togawa, M. Sato and T. Ohtsuki, "Maple: A simultaneous technology mapping, placement and global routing algorithm for FPGA's" *Proc. Int'l Conf. on Computer Aided Design,* November 1994, pages 156-163.

[12] S-C. Chang, K-T. Cheng, N-S. Woo, and M. Marek-Sadoswka, "Layout-driven logic synthesis for FPGAs," *Proc. 31st Design Automation Conf.,* June 1994, pages 308-313.

[13] J. Lou, A. Salek and M. Pedram, "An exact solution to simultaneous technology mapping and linear placement problem," *Proc. Int'l Conf. on Computer Aided Design,* November 1996, pages 583-588.

[14] K. Keutzer, "DAGON: technology mapping and local optimization," *Proc. 24th Design Automation Conf.,* June 1987, pages 341-347.

[15] M. Yannakakis, "A polynomial algorithm for the min-cut linear arrangements of trees," *Journal of ACM*, Vol. 32, No. 4 (1985), pages 950-988.

[16] G. Stenz, B. M. Riess, B. Rohflisch, and F. M. Johannes, "Timing-driven placement in interaction with netlist transformations," *Proc. the Int'l Symposium on Physical Design,* May 1997, pages 36-41.

[17] J. M. Kleinhans, G. Sigl, F. M. Johannes, and K. J. Antreich, "GORDIAN: VLSI placement by quadratic programming and slicing optimization," *IEEE Trans. on Computer-Aided Design,* March 1991, pages 356-365.

[18] B. Rohfleisch, B. Wurth, and K. Antreich, "Logic clause analysis for delay optimization," *Proc. 32nd Design Automation Conf.,* June 1995, pages 668-672.

[19] P. Abouzeid and K. Sakouti and G. Saucier and F. Poirot, "Multilevel synthesis minimizing the routing factor," *Proc. 27th Design Automation Conf.,* June 1990, pages 365-368.

[20] H. Vaishnav and M. Pedram, "Minimizing the routing cost during logic extraction," *Proc. 32nd Design Automation Conf.,* June 1995, pages 70-75.

[21] L. N. Kannan, P. R. Suaris and H. Fang, " A methodology and algorithms for post-placement delay optimization," *Proc. 31st Design Automation Conf.,* June 1994, pages 327-332.

[22] S-M. Liu, K-R. Pan, M. Pedram and A. M. Despain, "Alleviating routing congestion by combining logic resynthesis and linear placement," *Proc. European Conf. on Design Automation,* February 1993, pages 578-582.

[23] O. Coudert, "Gate sizing: a general purpose optimization approach," *Proc. European Design and Test Conf.,* March 1996, pages.