Design Planning Methodology for Rapid Chip Deployment

David E. Lackey
IBM Microelectronics, Essex Junction, Vermont, 05452
delacke@us.ibm.com

Abstract

Traditional IC chip design methodologies have adversely affected design turn-around time (TAT) by serializing many of the steps in the design flow. A design flow step either implements a design decision (the intellectual property or IP of the designer), or is a design processing and optimization step (raw processing of the design based upon requirements or optimization targets determined by the designer). The tasks involved in chip design are serialized when design decision steps are intermixed with steps involving design processing or optimization. The effect of a flow that is highly serialized, in this respect, becomes one of frequent designer-tool interactions and iterations.

This paper proposes a flow which provides for earliest completion of design decision tasks. This grouping of tasks is called Design Planning. Building on the latest advances in Placement-based Synthesis, the CPU-intensive tasks of Design Processing and Optimization are deferred and integrated into a final, one-pass process. Criteria for passage between the two major steps, identified as Design Planning and Design Processing and Optimization, are discussed.

Introduction

Achieving short time-to-market (TAT) continues to be a primary development objective of integrated circuit (IC) design. Once a functional design, often represented in a high level register-transfer level (RTL) form, has been completed and functionally verified, and once the objectives for logic path performance and silicon die size have been set, the schedule time to chip completion and manufacturing becomes the primary focus [1].

Traditional chip design methodologies tend to be serial in nature, as depicted in Figure 1. The function, defined in RTL form, is developed and verified using simulation and, more recently, formal verification methods. This is followed by logic synthesis, which maps the RTL into the gate-level circuits of the silicon vendor's technology. For logic synthesis, interconnect delay is modeled using wire-load models (WLM's). The designer iterates between synthesis, RTL changes, and choice of WLM's (with the aid of a floorplanning tool) until synthesis indicates timing closure at the proposed area [2]. Recently, synthesis tool vendors have added power optimization (minimization of power consumption), to this logic optimization process. Once the designer achieves closure in synthesis, Design for Testability logic is inserted into the design. This logic includes scan chains, and more powerful DFT extensions such as IEEE 1149.1 structures and Logic Built-in Self-Test. The designer defines and inserts clock (and other high fanout) powering networks. Next, sign-off tools, such as timing analysis, are applied to the design, to assess conformance to criteria for entry into the layout process [3].

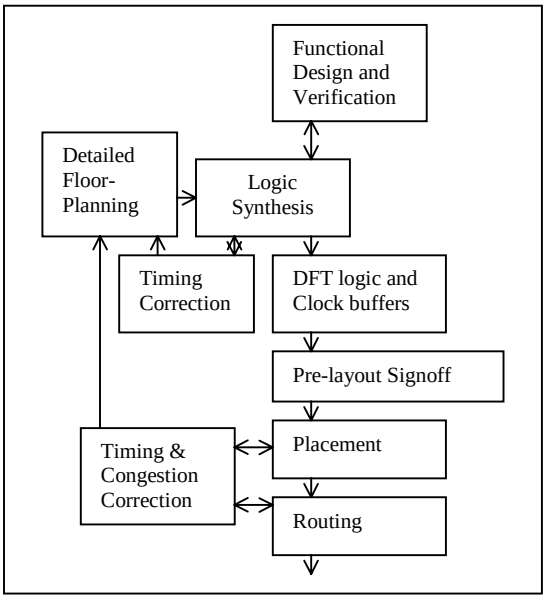

Figure 1. Traditional Serial Design Flow

Finally, the design enters the layout process. The physical designers use placement tools to achieve an initial layout, and measure timing based on estimated wire delays [4] (using methods such as Steiner estimation). A physical designer will resolve timing and congestion problems, using a variety of optimization tools and manual changes, until estimated timing and congestion appear to be good. Sometimes, these problems will require logic or floorplan changes by the logic designer, requiring a design handoff

between the parties. Then, further optimization tools are used to re-order the scan chains to minimize wire lengths and latch wire loads. The designer balances the clocks using a clock optimization tool, which re-organizes the connectivity amongst clock buffers and latches with respect to placement, and minimizes buffer loading towards an objective of minimal clock skew at each level of repowering [5]. Finally, the physical designer routes the design using a routing tool, and extracts parasitics for final timing analysis. At this point, further changes (ECO's) may be required to fix timing issues such as early mode hold problems, or to alleviate congestion issues. Again, this may require a design hand-off between the parties. These final fixes are affected using various optimization tools and manual means. Prior to release of the design to manufacturing, a test engineer produces a set of test vectors for the design, while a final set of manufacturability checks are executed.

In the traditional design flow, there is significant iteration between design engineering actions (in both the logical and physical design disciplines) and the CPU-intensive tasks of logical & physical design processing and logical & physical optimization.

Recent Improvements to the Traditional Flow

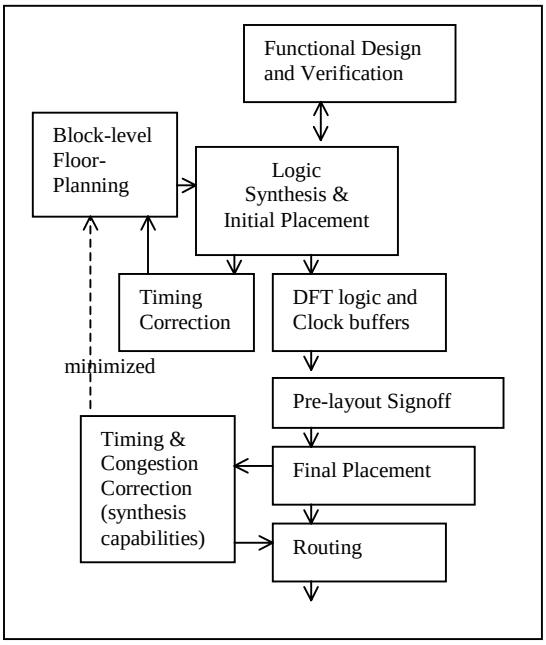

Figure 2. Placement-based Synthesis

Recent advances in the area of Placement-based Synthesis have improved this situation somewhat (Figure 2). Placement-based Synthesis techniques can be executed as an Early Timing Closure technique, in which the synthesis tool creates an initial placement that is simultaneously optimized with the logic design. This step appears after RTL completion in the design flow, and requires some initial block-level floorplanning. After this Early Timing Closure, the logic designer proceeds to DFT logic and clock logic insertion. Placement-based Synthesis techniques can also be executed as a Late Timing Closure technique by the physical designer. A fully placed, with an estimated, global, or detailed route, is optimized using a placement-based synthesis tool capable of localized changes to the logic and/or the placement, to fix localized timing or electrical issues [6].

While the Placement-based Synthesis improvements to the flow reduce the number of problems to be fixed by the physical or logical design engineer, there remains a number of required interactions by the design engineers throughout the flow, and required iterations between tools.

Design IP and Design Processing de-coupling

The level of designer interaction and tool iteration in the flow is in many ways tied to the inter-mix of two distinct types of design step:

Design IP: These are the decisions a designer must make. One area of such decisions considers the functional architecture and RTL implementation, and the methods used to functionally verify the RTL implementation. Another area considers the Chip's physical architecture. At this physical level, the designer makes decisions affecting major block placements against objectives for die size and area utilization, performance, and power consumption. Based upon metrics extracted during the design process, the designer reacts to problems in meeting these objectives, making appropriate changes in the functional design and/or the chip's physical architecture. Finally, the designer determines the logical clock domain organization and clock timing objectives, as well as the DFT objectives for manufacturing test and in-system test.

Design Processing: These are the tasks, often very CPU intensive, in which a design tool performs either optimization functions or inserts design objects or attributes into a design, or

extracts design measurements. Such tasks include logic synthesis, initial placement and optimization, DFT logic insertion and scan chain order optimization, clock powering insertion and clock network optimization, signal and power routing, parasitic extraction, test pattern generation, and checking tools for logic structure, placement, and wiring.

The serial nature of the current methodology flows can be reduced, by separating these two design step types. The steps involving design IP should be moved to the earliest possible point in the overall design flow. The steps involving design processing, especially those that are CPU-intensive, should be deferred to the latest possible point. In this paper, the steps involving functional design IP will be called ***Functional Design***. The steps involving Chip physical architecture, setting of clocking and DFT objectives, and early extraction of design metrics will be called ***Chip Design Planning***. The steps involving design processing will be called ***Design Processing and Optimization***.

An objective in deferring Design Processing and Optimization steps to the latest point in the flow is to achieve a single pass through these CPU-intensive operations. Requirements to achieve this objective are placed on Chip Design Planning, where exit criteria must be established to pre-determine success or failure through the subsequent processing flow. The capability must exist in Design Planning to capture metrics accurate enough to make such a prediction, and to drive Design Processing and Operation.

The Target Flow

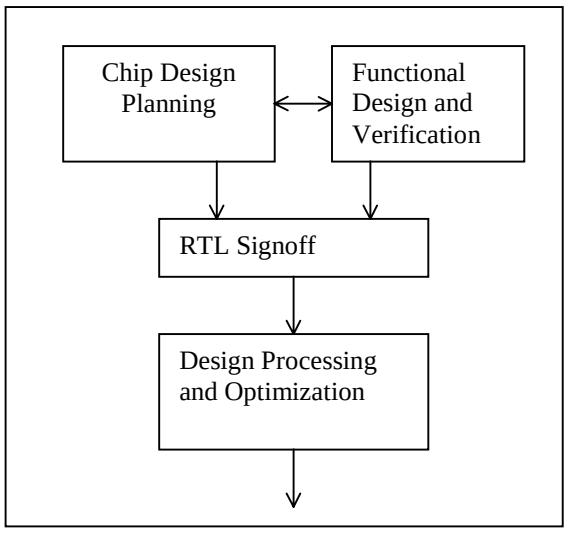

Figure 3. The Target Flow

A proposed methodology flow is shown in Figure 3.

Design Processing and Optimization

One method of determining the requirements of Chip Design Planning, is to first determine those steps that can be deferred to a fully-automated Design Processing and Optimization flow. This proposal should be based on capabilities that exist today in the Electronic Design Automation (EDA) industry, and enhancements that can be developed using known techniques. Then, after identifying the deferrable steps, the requirements for Chip Design Planning tools can be established.

For entry to Design Processing and Optimization, the fundamental requirements include the following:

- Functionally-verified RTL.
- Organization of RTL blocks, and large macros, to placement locations and physical block sizes.
- Chip I/O cell locations.
- Identification of interconnecting nets between blocks, and to chip I/O's.
- Exact or coarse identification of net entry/exit points on the RTL blocks.
- Determination that RTL for each block can be organized as specified within physical block sizes. This determination must consider cell area and utilization as necessary to meet the simultaneous objectives of timing, wirability, block input/output locations, placement porosity (localized area utilization limits) and power consumption.
- Modeling method and fanout limits to be used for RTL synthesis.
- Determination that interconnecting nets can meet the simultaneous objectives of timing and wirability, given the requirements of block-level input/output location and I/O cell location.
- Timing budgets for each block, and chip-level timing constraints.
- Macro-level DFT logic is included in the design, and start/end points for the scan chains are established, along with scan chain length balancing objectives.
- Clock (and high fanout net) structures, and their latency and skew targets, have been determined. Alternatively, for designs that will use skew to achieve timing objectives, a

clock delay variation is identified for each latch.

- Latch placement constraints, and structured data path placements, may be needed.

Other entry criteria, for which early measurement methods should exist, include chip-level voltage drop constraints due to on-chip and I/O switching (a function of performance), noise, and electromigration. Identification of power grid requirements for multiple voltage sources, power sequencing, and multiple site grids is also necessary.

Given these inputs and satisfied criteria, the following processing steps can be completed in an automated fashion:

1. Synthesize the logic of each RTL block based upon its timing budget, and area and power limits. Interconnect delay models were provided from Design Planning.
2. Insert the scan chain connections using the predefined start/stop points and the chain length balancing objectives.
3. Place latches and structured data paths as defined by Design Planning.
4. Reconnect scan chains to minimize the length of latch-output wires.
5. Insert and place clock buffers, based upon the structure defined in clock planning. The affinity of leaf-level clock buffers to latches is based upon latch placement, and affinity of higher level buffers to leaf buffers is based upon the latter's placement. If useful clock skew is a design objective for meeting performance goals, there may be delay cells that should be inserted and placed at the time of the leaf-buffer to latch connections.
6. Place the remaining logic gates of each RTL block, based upon the block shape and the locations (coarse or exact) of its inputs and outputs. This placement should be driven by the timing budgets, and driven using local congestion and placement density targets, and overall utilization targets. At this time, interconnect delay and congestion estimates should be based upon routing that has been estimated based upon the placement of the logic gates, and any temporary buffers (inserted earlier for fanout latency estimation) should be removed.
7. Place the top-level logic to meet chip-level timing targets, as well as congestion and placement porosity targets. Timing of paths through hierarchical blocks might be abstracted (hard macro approach) or calculated at the chip level.
8. Update timing budgets based upon actual placements.
9. Invoke localized placement-based synthesis optimization mechanisms against each block to complete the optimization process. If block-level inputs and outputs are coarse up to this point, create localized pin placements. Fix all electrical violations and legalize.
10. Repeat the prior step for the top-level.
11. Perform routing and final timing optimization.

Chip Design Planning

The requirements for entry into a one-pass & automated Design Processing and Optimization step, together with the steps performed here, identify many of the requirements for Chip Design Planning. Notwithstanding the vast amount and complexity of Functional Design IP, and necessary implementation and verification skill, the requirement for functionally-verified RTL is considered part of Functional Design and will not be considered an aspect of Chip Design Planning.

Chip Design Planning requires tools and a methodology capable of a number of significant functions.

RTL Analysis

1. Ability to load a chip-level set of RTL at a given hierarchical organization, identifying and analyzing global (inter-module) signals at any level of the hierarchy. Ability to reorganize the hierarchy, along with the contained RTL, based upon user reorganization of sub-modules. It is expected that reorganization will be based upon minimizing the number of, and the timing criticality of, global signals at certain levels of the hierarchy.
2. Ability to measure RTL based upon a certain pre-characterization of a target silicon technology. Measurements include area, power consumption, and timing for port-to-latch, latch-to-latch, and latch-to-port paths.
3. Block-level timing abstraction capability and time budgeting based upon chip-level constraints, including estimated slack on global signals, at user-defined levels of the hierarchy.

4. Block-level area and power estimation based upon the characterization described above.
5. Ability to identify clocks sources, the latches they drive, and to input required frequencies and relative phase relationships.

Synthesis Estimate Mode

In lieu of an RTL Analysis and technology characterization capability as described here, a logic synthesis tool might be set into an environment to perform a subset of the functions described above. Also, an early look at placement will be needed later in Design Planning, and thus will require a trial logic implementation. The answer to this need is called "Synthesis Estimate Mode," and might be set up in the following way:

1. Highest and lowest drive strengths of each cell family masked from synthesis mapping. This reserves such cells for later timing optimizations, area recovery, and power reduction, while minimizing structural perturbation.
2. Zero wire loads.
3. A pre-defined buffer size is made available to synthesis and a fanout limit is set, whereby the buffers that synthesis inserts will model latency for the buffer network needed eventually for high-fanout nets.
4. A pre-defined buffer is inserted at the entry and exit ports of each block.
5. Clocks are treated as idealized.

Any buffers inserted for the Synthesis Estimate Mode are flagged such that they can be deleted if these synthesis results ultimately become the target implementation.

Either using the RTL Analyzer or Synthesis in Estimate Mode, the designer can make an early determination, at this time, of performance capabilities, power consumption, and cell area budget. Further, block-level performance and power requirements, together with area, may be usable to estimate a block region's utilization target. Finally, a preliminary determination is needed of each block's size and shape, perhaps with a quick, unconstrained placement at the utilization target. For this latter requirement, an Estimate-Mode synthesis run is needed if not already done at this point.

Design for Testability (DFT) Planning

A design will require any number of DFT methods, both for chip manufacturing test as well as package, card, and in-system test. Methods may include:

1. Through-the-pins scan
2. IEEE 1149.1 boundary scan
3. Array Built-in Self Test (Array BIST)
4. Logic BIST

Scan chain connections through the latches will be made and optimized during Design Processing and Optimization. Further, the placement and optimization of on-chip DFT controllers (IEEE 1149.1 TAP, BIST controllers, etc.) will also take place at this later stage. However the logic connecting to scan chain endpoints (either related to BIST, I/O's or both), as well the DFT controller blocks, should be added to the chip logic during Design Planning [7].

High-level Floorplanning

Often, system packaging, or card-level, requirements will dictate exact, or at least coarse, placement requirements for chip I/O cells and connection pads. The floorplanning tool should provide robust methods for exact or coarse placements, relative placement grouping of I/O cells, and interoperability with circuit, image, and package level timing and power calculation tools. A coarse placement capability can be valuable for optimizing I/O versus cell and block placements. A legalization function is needed for I/O and related cells, to retain the coarse placement's timing and wiring objectives when locking these cells into legal I/O locations.

For a hierarchical design, key placement constraints need to be considered for the top-level and each block, simultaneously:

1. Points of entry for each block port from the top-level (pin locations)
2. Optimal size, shape, and aspect ratio of each block considering pin locations
3. Location and aspect ratio of each block and hard macro at the top-level.

Each of these issues might be addressed using a coarse-and-refinement approach.

1. Top-level considerations produce coarse block placements, using the RTL Analysis' size-and-shape estimates, and Hard Macro placements.
2. Top-level considerations produce coarse entry locations on each block. For a design employing multiple-block reuse, constraints are needed for these coarse locations such that the reused block will have, ultimately, the same placements for each instantiation.

Each block's size-and-shape and pin locations are finalized later, during placement of the block's internal logic. These finalizations are made within the coarse constraints established during the above top-level steps.

Clock Planning and Detailed Floorplanning

At the top-level and within each block, a quick placement is run, constrained by I/O and pin placement, and with objectives to minimize a function (timing-based) of wire length, and the designer's objectives for localized placement.

Latch locations that result from this placement are indicative of the required area of their containing clock domain, and derive optimal location of large clock buffers needed at the root of the domain's repowering network. A Clock Planning tool will use this latch area, along with the performance requirement and target clock latency, to determine the structure of the tree. This includes the sizes and placement (relative to this latch area) of the large root buffers. These large clock buffers should be placed.

For this initial trial placement, a mechanism whereby timing tracks well to a function of wire length is preferable for predicting later optimization success. A quadratic placement algorithm is one example of this.

At this point, coarse placements exist for I/O's, block pins, and latches. This initial trial placement, which also includes the remaining logic, should also provide indication whether any reshaping improvements, for the blocks, should be made. The designer should now provide the target block shapes, and use the trial placement to finalize block pin locations, and finally I/O and related cell placements. Alternatively, the block pin locations themselves might be left coarse at this time, and finalized during Design Processing and Optimization. Latch placements need not be legalized at this point. The remaining random logic cells can now be unplaced, and replaced based upon the finalized block shapes, pin locations, and I/O and related cell placements.

Design Planning Metrics and Data Extraction

The logic, at this point, contains temporary buffers inserted previously by Estimate-Mode Synthesis, in response to high fanout nets and/or electrical correction by the synthesis tool.

Using the results at this point, the design should be measured as follows:

1. Static timing based upon placement-based wire estimation, gate pin capacitance, and gate delay, at the worst case operating point. Clock network timing should be idealized based upon the latency targets given to Clock Planning. Thus, early-mode timing within clock domains need not be assessed at this time. A late-mode margin should be applied for each clock domain, in addition to the design cycle time and asserted arrival times. This is the sum of the following factors: a measure of uncertainty in tool-generated clock latency at any latch in the domain, PLL jitter, and any additional margins desired to trade-off performance and TAT.
2. Localized wiring congestion (Steiner-estimated) and placement density maximums, as well as global utilization targets.
3. Power consumption using statistical or simulation-extracted switching data.
4. Scan-based test coverage using an ATPG tool. This step should be done by this time, but might be accomplished as early as the RTL analysis if capable, or with first Estimate-mode synthesis results. Some ATPG tools contain an assumed-scan capability, which is necessary at this time since internal scan chains have not yet been created.

Problems encountered at this point should be corrected. Most of the design parameters measured at this point were design planning objectives earlier in this process, thus the designer's objective is to guide his Design Planning activities so as to minimize the shortfalls to be discovered at this time.

Once the metrics meet the design requirements, the designer uses the Design Planning tools to extract the following data to guide the Design Processing and Optimization process:

1. I/O cell placements
2. Block shapes and placements
3. Large clock buffer placements
4. Coarse or exact block pin placements
5. Coarse latch placements or latch cluster placements.
6. Placement utilization factors and localized placement density and congestion maximums.
7. Block-level time budgets.

8. Wire loads for final RTL synthesis: zero-loads within synthesis blocks, and Steiner-estimated loads (with tools capable of extracting loads for actual routes if available) for synthesis block interconnections.

Conclusions

Most aspects of Design Planning are able to proceed in parallel with Functional Design, whereby checkpoints in the Functional Design process can be used to spin-off design versions for Design Planning. As such, with the exception of final metric analysis and data extraction, Design Planning is not in the serial schedule path.

Success in Design Planning, both in terms of criteria achievement as well as delivery of reliable driving data for Design Processing and Optimization, provides the best possible chance of a push-button, one-pass process through Design Processing and Optimization.

The primary benefit of this methodology is most rapid deployment of chip designs once the RTL-based functional design and verification have been completed. However, Design Planning provides for earlier and more accurate predictability of final design metrics, leading to possibilities for improved performance, smaller die size, and reduced power consumption.

Acknowledgments

The author acknowledges the following individuals, whose efforts form the basis of many of the points made in this paper:

- Tom Bednar, John Cohn, George Doerre, and Juergen Koehl, for contributions to the design philosophy described by this paper.
- Joe Morrell and Jose Neves for approaches to hierarchical design.
- Bob Proctor for his efforts in I/O planning.
- Mark Aubel, Keith Carrig, Daniel Menard, Harold Reindel, and the IBM EDA Clock Optimization team for their efforts in clock design.
- Mark Aubel and the IBM EDA and Research PDS team for their efforts in placement-driven synthesis, and Darell Whitaker and Jenny Chu for their efforts in supporting third-party vendor physical synthesis tools.
- Elizabeth Bouldin, Vivek Chickermane, David Litten, Steve Oakland, and Lori Smudde, for their work in DFT planning and automation.

References

(1) (6) D.Lackey, “Applying Placement-based Synthesis for On-time System-on-a-Chip Design,” Proceedings of the IEEE 2000 Custom Integrated Circuits Conference (CICC), pp.121-124

(2) J.Czilli, A.Nordstrom, “Synthesizing a 650K Gate Deep Submicron ASIC,” 1997 Synopsys Users Group (SNUG ’97)

(3) (5) (10) J.Engel, T.Guzowski, A.Hunt, D.Lackey, et al., “Design Methodology for IBM ASIC Products,” IBM Journal of Research and Development, Volume 40, No. 4, 1996, pp.387-406

(4) C.Chen, Y.Chen, D.Wong, “Optimal Wire-Sizing Formula under the Elmore Delay Model,” Proceedings of the 1996 IEEE Design Automation Conference (DAC), pp.487-490

(7) V.Chickermane, D.Lackey, D.Litten, L.Smudde, “Automated Chip-Level I/O and Test Insertion Using IBM Design-for-Test Synthesis,” IBM MicroNews, Second Quarter 2000, Vol. 6, No. 2