

# Accelerated Optimization with IC Compiler II

## Rethinking Optimization for Tomorrow's Challenges

### Author

**Charles Hsiao**

Staff CAE, Synopsys

### Introduction

Ultimately, the challenge of physical optimization is to provide design convergence and closure as a design moves from the logical realm into the physical world.

Technology scaling, in as much as it drives the density demands required of Moore's law, comes with growing challenges of increasing interconnect delay. As we transitioned through maturing nodes, cell delay was the dominant factor. However, as wire cross sections have shrunk with each new node, the associated wire resistivity has grown exponentially.

Therefore, today's designs are becoming increasingly "buffer heavy." It has been reported that in newer technologies, it is not unusual to see over 25% of all cells merely existing to move signals around the device in a timely manner. This trend is not expected to slow down even with the introduction of higher power devices such as FinFETs; their higher drive being offset by more highly resistive materials required for multiple patterning, as well as larger dies—thus longer nets—for growing designs.

Mitigating this increased interconnect delay is thus the key gating factor for successful closure.

Other key considerations including power management, design routability and signal integrity are also becoming increasingly demanding factors that need to be addressed efficiently during optimization. Trying to fix these issues late in the flow can often result in either extremely poor or indeed a total lack of design convergence.

As mentioned above, global net length and the growing buffer requirements is a function of increasing design size. Another challenging aspect of growing design size is that of capacity management. Design growth necessitates the need to increase block sizes and this has to be absorbed by the optimization engine without unduly affecting block throughput or memory requirements. This duality of throughput and capacity thus need to be addressed wholly and holistically if a scalable solution that can handle tomorrow's designs is to exist.

### Delivering Leading Optimization: Advancing the Paradigm

As discussed in the introduction, physical optimization is a complex, multi-objective problem that places large demands computationally on EDA software. The classic approach to handling this large computational demand has been the deployment of multiple algorithmic and heuristic trials that increase in their accuracy as the flow progresses. To ensure convergence, regular timing updates need to be made in order to qualify the efficacy of each of these "moves" or "tricks." It is in these timing updates, many of which are effectively discarded, where significant runtime is consumed.

Also, when considering small windows into the circuit, it becomes easy to negatively impact prior improvements in previous windows for the same, or even different, cost objectives. The sequential nature of applying these costs can also either leave significant QOR on the table or local maxima reached that are not optimal more globally. This often results in a “ping-pong” effect where multiple passes of optimization need to be performed, negatively affecting throughput and the timing convergence landscape.

IC Compiler II’s architecture has been developed to address these and many other shortfalls in classic optimization methodologies. Fundamentally, IC Compiler II’s analytical physical synthesis (APS) engine works on a global scale when dealing with regions of interest. In this way, it partitions optimization opportunities into “logic-level slices” across the timing graph thus enabling it to identify more optimal maxima in the relevant, physical locale. IC Compiler II’s timing infrastructure and its integration natively across the full datamodel allows for a tighter and more expansive coupling between optimization and timing updates. Expanded, sub-circuit timing graphs are efficiently presented to the optimization engine thus delivering fast, incremental updates that allow for visibility of potential optimization conflicts.

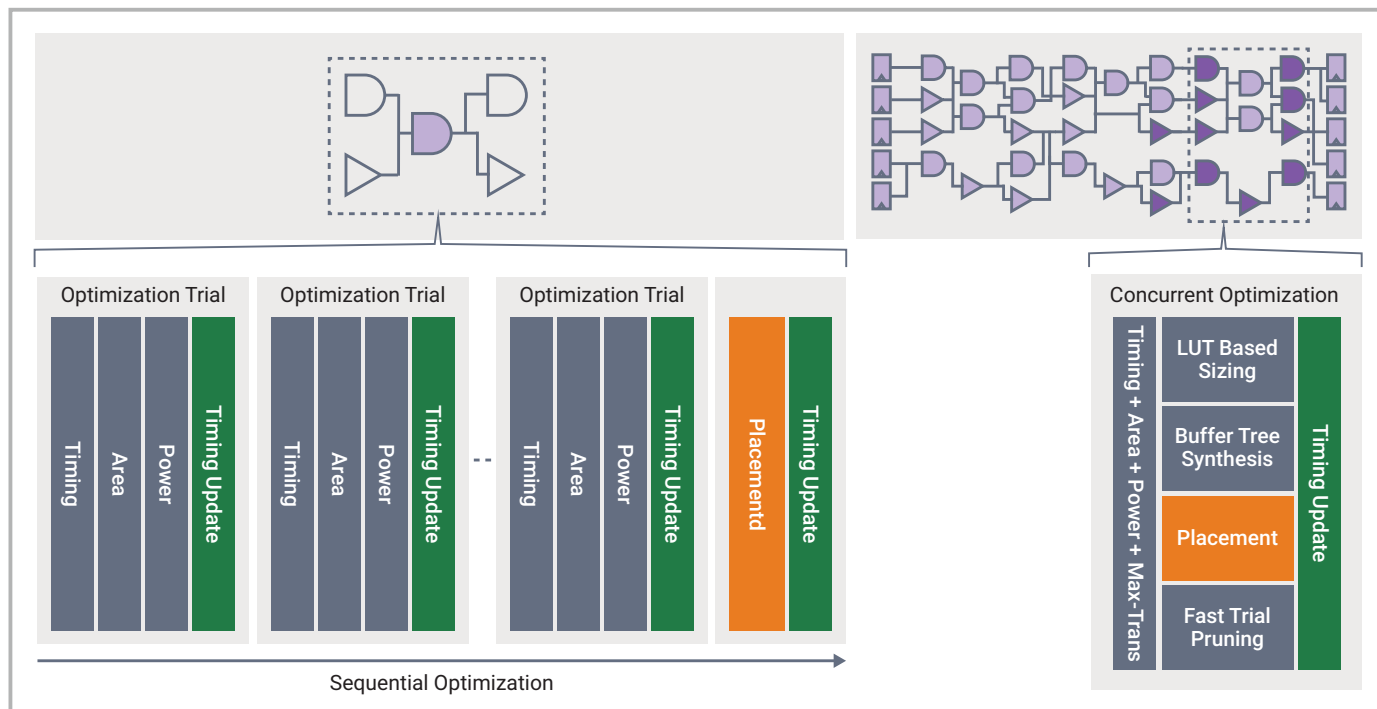


Figure 1: Comparison of Classical Optimization and IC Compiler II’s Optimization Paradigm

Figure 1 shows the comparison between classical optimization strategies and IC Compiler II’s analytical physical synthesis (APS) approach. In APS, all applicable techniques for optimization—including analytical buffer insertion—are natively parallelizable across multi-core machine architectures. This inherently delivers significant throughput advantages and expands the methods that can be applied for the most convergent optimization.

The global nature of the sub-circuit presented to the optimization engine additionally provides for a more globally optimal solution when deploying the many strategies such as re-synthesis, re-buffering, cloning, decomposition, re-ordering.

Another key aspect of the APS engine is that of concurrently performing placement during logical transformations. Cell movement can significantly affect load capacitances and negatively affect routability. Performing these steps in unison ensures that the engine sees the true effect of the currently proposed changes to the circuit.

As a design moves through optimization and ultimately through the flow, broad stroke, lower accuracy transforms, need to proceed to higher accuracy design modifications. Whereas early in the flow Steiner estimates are sufficient, eventually these delay and load estimates need to become more refined.

IC Compiler II's APS engine delivers flow convergence through the timely application of three, key technologies:

- Advanced, virtual route based net estimation technology
- Integrated access to Zroute's global route (GR) engine
- Low overhead, high accuracy, timing model application

Integrating Zroute's global router into the pre and post-route optimization flow ensures that congestion, routability and wire delay is accurately and consistently modeled and captured. This is increasingly important in designs with highly fragmented floorplans where likely topologies of nets need to be seen as early as possible. Similarly, when heterogeneous layers stacks exist, ensuring that probable layer assignment is correctly captured helps to deliver fewer surprises and more optimal resource usage during the implementation flow.

## Pre-Route-Optimization Convergence: Going Beyond Placement

The optimization techniques that have largely been discussed till now have focused on physical placement and its subsequent optimization. However, what is becoming a more challenging aspect of physical design is that of delivering convergence during and after clock insertion.

Clock insertion may not be able to deliver or may not be geared towards zero-skew trees—something that early physical optimization often assumes. For example, low-power trees or low-power branches of trees often have competing goals to zero-skew trees and thus ensuring that physical optimization understands these needs is paramount.

Indeed, the placement of clock gates and their effects on on-chip-variation (OCV) divergence and enable timing, if not captured early, will often lead to un-closable designs.

IC Compiler II's pre-route paradigm has been architected to bring together these normally disparate steps together into a more efficient, more convergent, pre-route stage. IC Compiler II's concurrent pre-route flow comprises the following major steps:

- Industry leading physical placement
- Early, fast, clock tree synthesis (CTS) – identifying enable timing estimates and insertion latencies
- Early optimization with advanced ICG management techniques
- Focused placement and physical optimization—targeting optimal ICG topology generation
- Advanced clock tree synthesis and optimization
- Concurrent, physically aware clock and data optimization
- Final placement and closure

How these major stages interact is depicted in figure 2. It can be seen that significant overlap exists between all the major steps in the flow with optimization being a key contributor to the entire stage. By deploying a common optimization infrastructure across the entire process, collective goals are met in a convergent manner. All timing, power and area metrics, as well electrical requirements such as noise, slew and transition fixing are done simultaneously for both clock and data path signals. This helps ensure that they are addressed at the optimal location within the circuit.

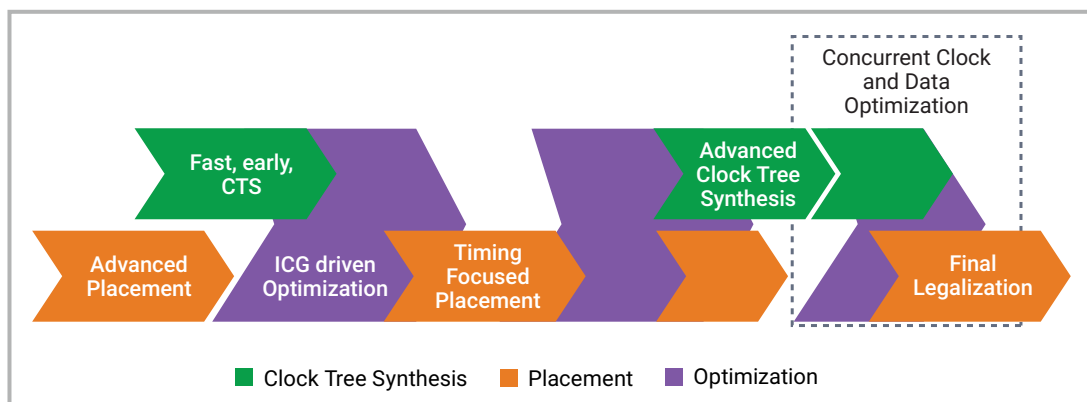


Figure 2: Convergent pre-route optimization flow

## Speeding Optimization: New Infrastructure and Technologies Speed Turnaround Time

Some of the key, quality-of-results focused technologies in IC Compiler II have previously been described. However, the underlying architecture of the datamodel needed to be re-thought if the capacity and throughput demands were to be met.

IC Compiler II's datamodel links to a new library system where all physical, logical and timing data comes together in a compact and more efficiently managed format. Data redundancy is removed from classical library implementations and this delivers massively reduced in-memory and on-disk representations of the data. The natively multi-threaded datamodel is thus able to present all of the relevant information to each of the application engines, extremely quickly.

IC Compiler II also deploys a single, cross-flow timer that helps deliver best-in-class timing convergence. Its natively multi-threaded, multi-scenario architecture provides for broad-flow, comprehensive, multi-mode-and-corner analysis thus requiring fewer or indeed a removal of ECO closure loops.

Figure 3 shows a comparison of IC Compiler II's concurrent pre-route optimization to classical optimization flows and the associated memory and runtime improvements this delivers.

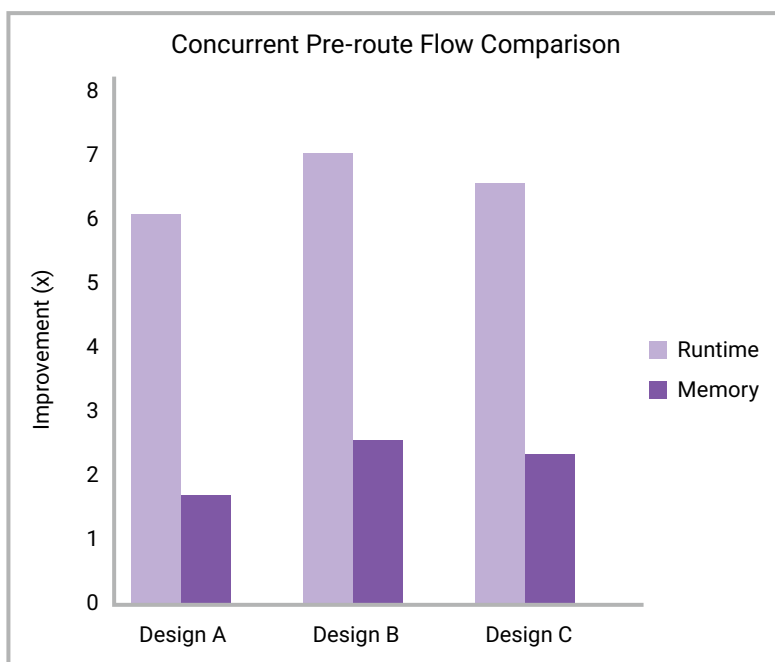


Figure 3: IC Compiler II vs. classical performance comparison

## Summary

Efficient optimization is a necessary yet challenging aspect of the physical implementation flow. Newer nodes and growing designs are all conspiring to place growing demands on this already difficult area of efficient IC design.

IC Compiler II and the underlying physical optimization engines have been re-thought and re-architected to address these growing challenges and to set the stage for the future. Numerous new technologies have been deployed including analytic physical synthesis, concurrent pre-route optimization, advanced design closure methodologies as well as optimizable physical abstracts for efficient hierarchical design closure.

Coupled with a re-thought data model, IC Compiler II's optimization engines deliver leading quality of results, fast and predictable design convergence with broad, customer-validated throughput benefits.

To learn more about IC Compiler II visit: <http://www.synopsys.com/ICCompilerII>