WHITE PAPER

GLOBAL SYNTHESIS FOR DESIGN CLOSURE

THE IMPACT ON PHYSICAL QUALITY OF SILICON

cādence™

OVERVIEW

Chip complexity has grown more than 1,000 fold since the last major innovation in front-end design technology. Such growth has intensified the challenges that design teams face when producing silicon with more than 700M transistors. More challenges will follow for teams working on SoCs of similar complexity. Back-end retooling issues for designs at each new process node are studied widely as suitable EDA products become available. Meanwhile, the importance of the front-end in design closure is receiving much less focus. Some experts suggest that the front-end designer may no longer be responsible for design performance or power consumption, and that the end results are determined completely by the back-end process. This paper dispels that notion, along with others that surround physical synthesis.

The paper explains why the front-end designer must create the global logic structure that feeds the design closure process. In addition, the elements of high-performance design are reviewed, including the globally based optimization algorithms necessary for facilitating design closure. Numerical data is presented to support the suggested rules for creating a design. The premise of this paper is that the front-end design process now represents the weakest link in the implementation chain for complex designs. There is a clear need—and thus the opportunity—for a new generation of front-end solutions to tackle the challenges of more than a decade of process technology evolution.

BACKGROUND

The implementation of chip design progresses through a chain of steps guided by tools, methodology, experience, and design goals. This chain delivers an implementation that is only as good as its weakest link. Ideally, the process technology—and only the process technology—would be the limiting factor in the successful completion of a chip implementation. However, in the real world, tools are most often the source of project difficulties. By focusing on improving the weakest tools, we can greatly improve the design process. Using the best-in-class tool at each stage of implementation can ensure tapeout success.

Over the past 15 years, fundamental changes in each generation of back-end implementation tools have occurred. For instance, in cell placement, there was a shift from pseudo-random, simulated annealing-based tools to a new generation based on quadratic constructive algorithms. The key to success of these placement algorithms is the combination of bipartitioning and global optimum placement—a method that uses a top-down approach until all modules are placed. Today, all commercial placement tools use this technique, which represents the biggest breakthrough in standard cell placement technology in the last 15 years.

The transition of the majority of path-delay shifting from the active devices to the interconnect between devices has forced significant refinement of the back-end tools to address the timing problem for closure. By incorporating incremental optimizations in sizing and buffering, placement tools can now refine the design using actual physical information. In contrast, no corresponding breakthroughs or innovations have occurred for front-end implementation tools. In fact, R&D spending for the back-end has been at the expense of the front-end.

As the weak link, the front-end design process needs to be reevaluated and given major focus to remedy this growing deficiency. Why the focus on the front-end now? Many 45nm processes are moving into production. Initial results have been reported for the first 22nm processes in R&D lines. For a perspective on the enormity of the complexity that designers soon will be facing, ponder the following numbers:

Compared to 90nm circuits, 45nm circuits have:

- 4 times more gates per unit area
- 2 times the performance improvement
- 0.5 times the power dissipation per MHz per gate

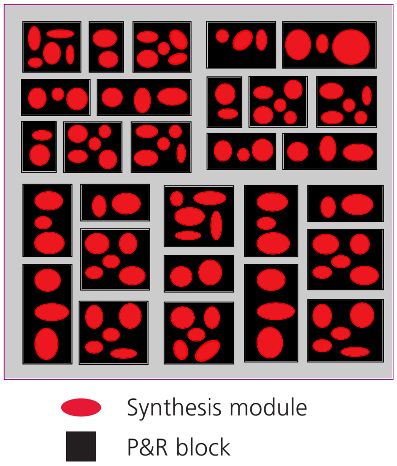
That means that for a simple process shrink, assuming gate count and performance are kept the same, power can be halved. But if you take advantage of the increase in gate capacity and/or performance, then power consumption will most likely increase by a large amount. Therefore, trading off area vs. performance vs. power becomes even more crucial.

Integrated circuits of 45nm geometry have resulted in new levels of density and complexity:

- 10 x 10mm die size: 148M gates
- 15 x 15mm die size: 333M gates
- 20 x 20mm die size: 592M gates

LIMITATIONS OF TODAY'S SOLUTIONS

Many factors explain why current front-end synthesis practices are starting to fail. One of the most obvious is the growing project complexity caused by synthesis methodologies dictated by the limitations of the old tools. To maintain runtimes at reasonable levels, achieve satisfactory quality of results (QoR), and control the debugging process, design teams have been using an exaggerated partitioning approach to synthesis. Developed under original capacity limitations, the old synthesis tools have imposed complexity in design data management—complexity that is growing at a pace consistent with Moore's Law. *Figure 1* shows the complexity management problem that design teams face when working on high-performance chips that are just a few million gates. At this time, scaling existing design practices to over 500M gates is simply not practical.



For a 20M gate chip:
Four blocks of 5M chiplets
(1.5M instances for physical design)
200 blocks of 100K instances for synthesis

For a 50M gate chip:
Ten blocks of 5M chiplets
(1.5M instances for physical design)
500 blocks of 30K to 50K instances for synthesis

Figure 1: Complexity management problem

Another problem with existing front-end synthesis solutions is that they achieve much of their performance optimization by sacrificing time and results. Squeezing the last bits of performance out of a suboptimal global logic structure using many hours of local sizing and buffering operations simply wastes time. The final refinements to sizing and buffering can be made only by adding adequate granularity of physical detail. Refinement into a local minima actually reduces the likelihood of finding an optimal result. The optimization approach is limited in part by its lack of capacity and its local nature.

In addition, the core algorithms supporting existing tools behave negatively with respect to timing closure. In particular, netlists created with the old solutions commonly have congestion problems and suboptimal performance. When examining designs that are experiencing difficulty in closing, we have observed that the narrow selection of library components and the accompanying lack of critical signal isolation intensify transition issues between the front-end and back-end design environments. This behavior is attributable to the local nature of the optimization found in the current tools.

A CHANGE IN UNDERSTANDING

Some observers have suggested that the front-end designer may no longer be responsible for design performance and power, and that optimizing for these goals is completely a back-end process. Designers have stated that "the netlist doesn't matter; the delay is all in the wires." This may reveal a lot about how physical synthesis tools have been marketed, but does not contribute to finding a solution to the actual challenge. Back-end designers become frustrated when they suddenly find themselves getting lower-quality netlists because the front-end designers know that they will use incremental physical synthesis. Microarchitecture and global logic structure cannot be fixed with incremental optimization techniques.

Further, the scope of the optimization transformations must be narrowed for a design to close. As the designer gets closer to the end of the design process, changes should become smaller and more local in nature. Otherwise, the upheaval can create ripples throughout the entire design and be the source of endless iteration. Big problems need to be fixed at the front-end, while the back-end is suitable for smaller, local changes (such as sizing and buffering).

It shouldn't be surprising that designers cannot simply feed poor-quality data into an optimization process and expect to get perfection from it. The strongest control that designers have over the performance of the design is still the RTL source code. Even the best synthesis tools cannot overcome a bad microarchitecture. Further, poorly optimized or non-optimized netlists serve as inadequate input for placement. Bottom line: The RTL and the netlist it produces matter. Therefore, we are focusing on globally based optimization algorithms for new synthesis solutions to the complexity that designers now face (*see Table 1*).

Feature	Old	Globally based
Gate capacity	0.3M gates	4M gates 32-bit, unlimited 64-bit
2M-gate runtime	100+ hrs.	8 hrs.
2M-gate memory	10+ GB	1.5 GB
Lines of script	3,000	300

Table 1: Projection of how new global synthesis tools may handle tomorrow's complexity

THE KEY ROLE OF SYNTHESIS IN DESIGN CLOSURE

The RTL code that the design team creates serves as the starting point for design implementation. However, there are theoretical limits to the level of improvement possible within the confines of the microarchitecture described in the RTL. Optimizations such as retiming can mitigate a suboptimal local microarchitecture by moving logic across registers. But the majority of the design community has not yet embraced even such small microarchitecture optimizations because of the verification problems that ensue.

Given that an RTL synthesis tool must operate within the confines of a microarchitecture, the highest level at which this tool can make an impact is on the global logic structure for the design. In terms of optimization and the optimization objective function, the global logic structure represents the location of a design structure that will produce the best quality of silicon (QoS) throughout the remainder of the implementation and refinement optimizations. *Figure 2* depicts the location of a global optimum "valley" that can be subsequently refined with local optimizations as downstream implementation detail is refined.

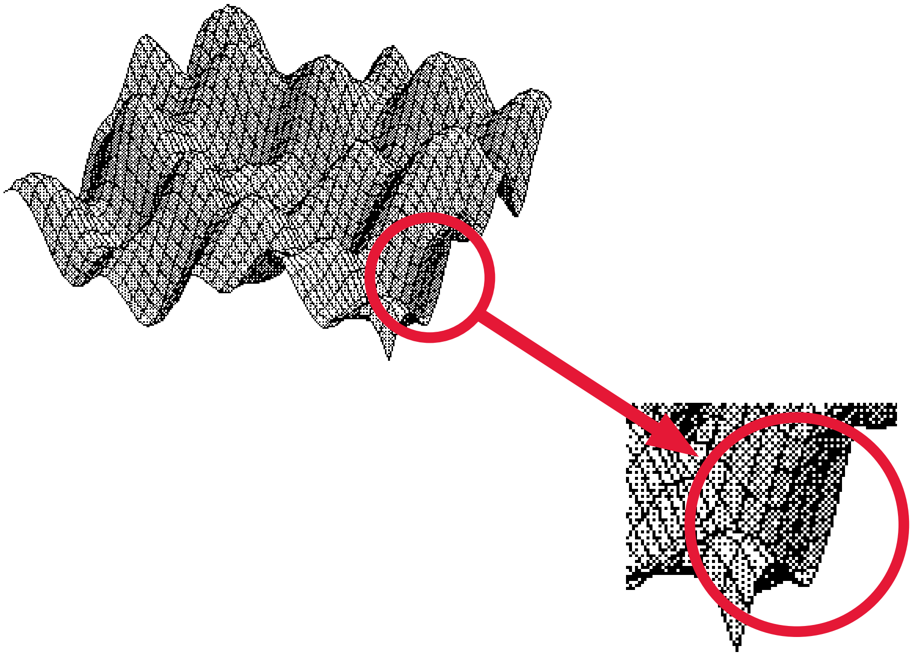


Figure 2: Global to local optimization refinement

A second critical role that synthesis plays in design closure (which many ignore) is the integral relationship of timing, area, and die size. In customer-owned tooling (COT) design flows, teams set die size according to the relative importance of cost, performance, and level of automation achievable. A low utilization value enables a high level of automation (e.g., 100 percent automatic completion of placement and routing). However, the tradeoff for this automation is in die size.

Design teams constantly juggle to maintain the highest utilization value possible with a level of automation suitable for the target schedule. Numerous studies have shown the effects of shifting utilization values upward and downward. The common result, regardless of the target place-and-route tool suite, is a very steep utilization "wall" (*see Figure 3*). The wall is between the "low zone," where the highest levels of automation are possible, and the "red zone," where the tools break down and there may be no feasible solution. The wall itself is called the "red zone," and is usually bound by a utilization variation from 3 to 5 percent. Thus, 3 to 5 percent in cell area often represents the go, no-go difference for applying a high degree of automation for a given die size.

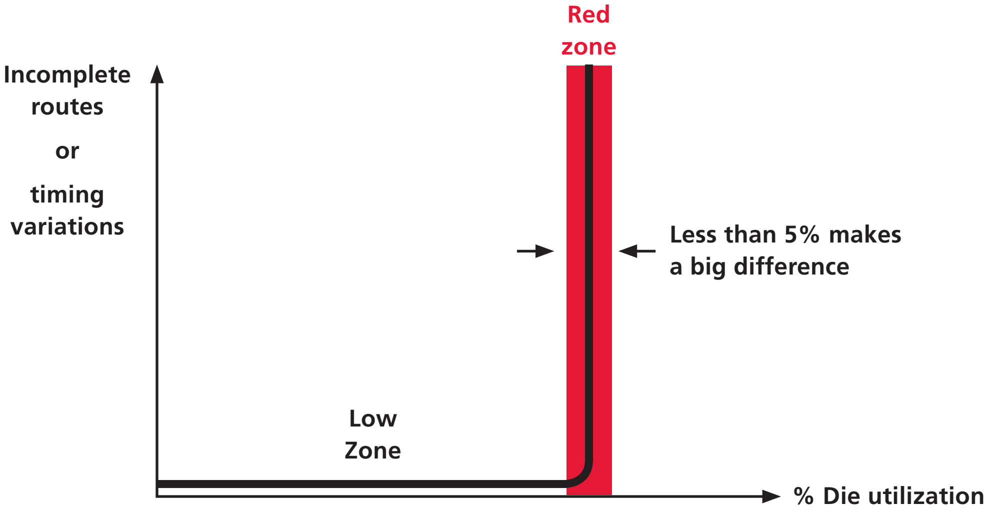


Figure 3: Utilization wall

As utilization approaches the red zone, meeting timing closure goals becomes increasingly difficult. When utilization—and thus cell density—increases, congestion in interconnect grows. The result is a vicious cycle of more detoured routes, leading to more long wires that require the insertion of more buffers, which increases the cell density, and so on (*see Figure 4*).

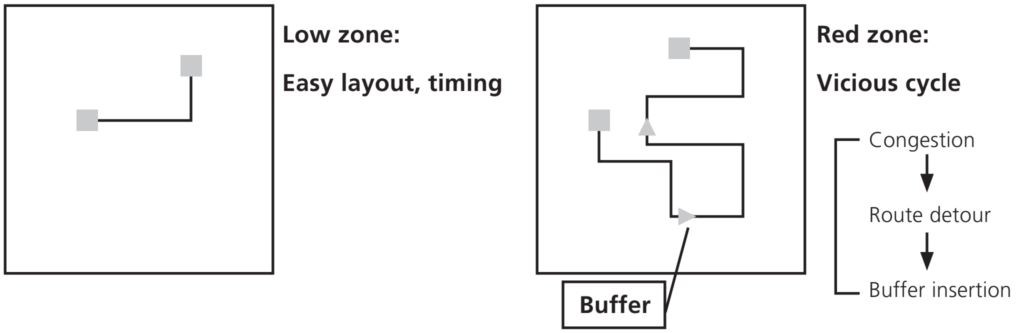


Figure 4: Low zone, red zone

Most design groups would like to operate as close to the wall as possible without hitting it. To do so, every percentage of cell area savings counts.

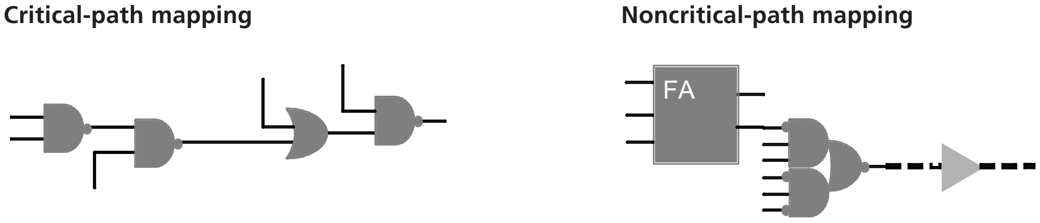
The third critical role that synthesis plays in design closure is in terms of power. Traditional methodologies have treated power optimization as a back-end problem. As with timing, performing cell swapping at that point is not enough. To reduce dynamic power, logic structures must be created to minimize switching activity. Leakage optimization requires tradeoffs between power, timing, and even area. This is when the isolation of timing-critical signals and the availability of a wide selection of cells become even more important. At 65nm, the difference between high- and low-voltage threshold cells is about 50 percent in terms of delay, but about 30x in terms of leakage. A good netlist minimizes the amount of timing-critical logic that may need to use higher-drive or low-voltage threshold cells in physical design closure.

TOWARD AN IDEAL SYNTHESIS TECHNOLOGY

An optimal synthesis technology must handle the massive capacity demanded by rising chip complexity (gate count). It must take into account a fundamental shift in optimization technology to build global logic structures conducive to achieving rapid closure throughout the remainder of the implementation and incremental optimization process. A logic structure for timing closure has been a conceptually difficult issue because some ad hoc measures of netlist “goodness for place and route” used in the past are no longer valid. In particular, our data shows that isolated use of metrics such as number of cells and nets is misleading at best (see “Promising Results for Global Synthesis” section).

The treatment of critical- and noncritical-path cell selection exemplifies the types of differentiated optimization that can drive implementation of a global logic structure that has the best chance for first-pass closure. On critical paths, it is of paramount importance that the most time-critical signal be pushed as far forward in the logic computation chain as possible. Also crucial, the signals must be isolated as much as possible from loading by noncritical paths.

The result is that critical paths will contain mostly gates with a small number of inputs with low fan-out along the path. When the critical signal arrival times are equal, more gates with larger fan-in will be used. Note that this gate formulation is not the same as merely exploding complex gates into multiple levels of logic, which would only increase the total interconnect delay along a critical path. Instead, the formulation is a different global logic structure that drives the gate selection. Additionally, most cell libraries contain a richer set of sizing options for smaller cells. This provides a built-in self-buffering property for downstream incremental optimization tools, including modern placement tools.


Critical-path mapping

Noncritical-path mapping

Figure 5: Strategy for critical vs. noncritical paths

For noncritical paths, the global logic structure needs to balance the gate selection used on the performance-critical portions of the design. Lowering of instance and net count helps reduce the optimization chores of the downstream tools. And mapping these structures to high-voltage threshold cells helps keep leakage power in check. Thus, cells such as full adders (FA) and compound combinational logic functions will be abundant on paths with timing slack. Because these cells typically have fewer sizing options in the cell library (usually just a high-drive and a low-drive version), buffer insertion optimization will more likely be used to augment such complex cells when they are driving high fan-outs or long interconnects. *See Figure 5*.

PROMISING RESULTS FOR GLOBAL SYNTHESIS

It is impossible to offer formal proofs of the assertions discussed in this paper. However, mounting experimental evidence supports these postulates. The following highlights of results come from simple case studies of production chip design projects utilizing Encounter® RTL Compiler global synthesis.

One of the more critical aspects of scaling design methodologies for 65nm and denser process generations is the change to optimization algorithms that grow linearly (at worst) with design size. The nonlinear characteristics of the old algorithms have dictated the use of the hyper-partitioned design styles common in most multimillion-gate projects today. Encounter RTL Compiler's global synthesis algorithms exhibit very good memory and runtime behavior over a broad set of module sizes. *See Figure 6*.

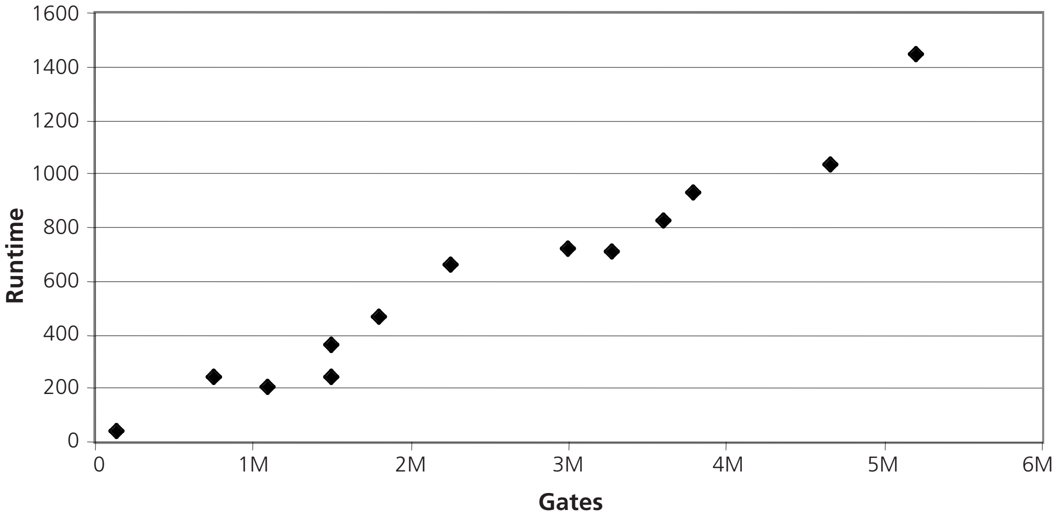


Figure 6: Global algorithm runtime, as a function of design size, demonstrates linearity

Another aspect of algorithm performance that needs to be examined for 65nm and smaller geometries is characteristics over the number of cells in the library (*see Figure 7*). Many users of the old algorithms purposely reduce the effective cell set (e.g., with "don't use this cell" directives) to control QoR problems and runtimes. Studies of Encounter RTL Compiler's global synthesis algorithms show that they can effectively take advantage of sets of more than 10K cells, as linear runtime and memory usage increase. With today's multi-threshold voltage libraries that need a large number of cells to effectively leverage process innovation, use of large cell sets is very important.

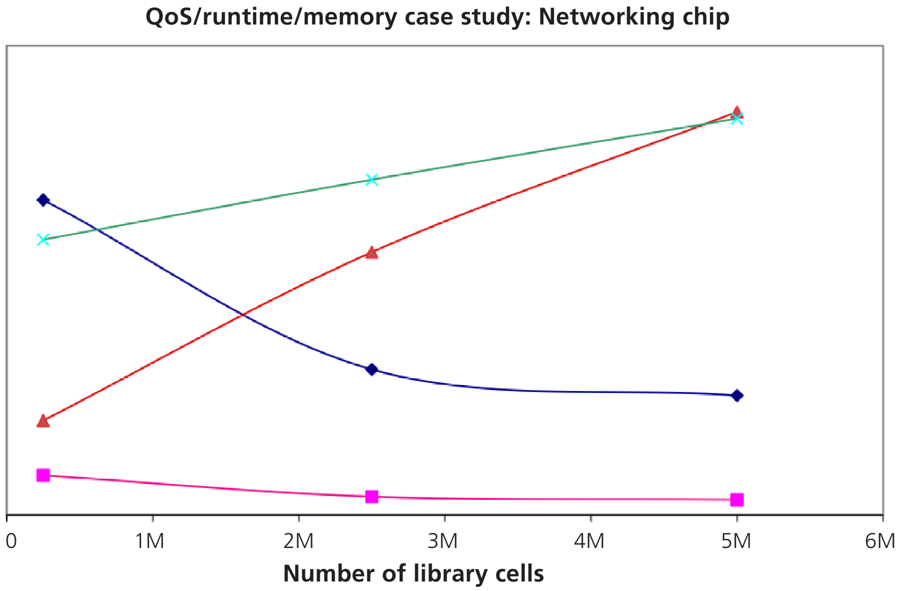


Figure 7: Library cell count study

The first design-project case study was based on a large (10M+ gate) networking chip targeted for the high-speed router market. The design flow for the chip used a COT model, and the target foundry was one of the world's largest. The performance target for the chip was set at 400 MHz. In this case, the design team had been iterating from RTL changes, gate instantiations, and placement, routing, and convergence tools for two months, with performance stuck at 326 MHz. The team had reached a local minima for that design. The RTL was then taken into Encounter RTL Compiler global synthesis. These new algorithms found a solution that ran at 417-MHz pre-layout timing. The team ran the resulting netlist through the placement, routing, and convergence flow in a single pass. The design exceeded the 400-MHz performance goal. *See Table 2*.

Metric	Old synthesis	Global synthesis
Time-to-result	2 months	2 days
Post-layout timing	326 MHz (violation)	400 MHz (met timing)

Table 2: Design-project study 1—performance results

The second design-project case study was a processor for a wireless device, also using a COT design flow. In this case study, the design team met their QoS goals in pre-layout analysis after about three weeks of iteration in synthesis. However, when the team ran the netlist through physical timing closure, leakage power increased by an unacceptable amount.

On the most critical block, of about 400K instances, the physical synthesis was struggling to meet timing with the existing synthesis netlist, so it needed to use a large amount of low-voltage threshold cells. The resulting leakage power was 50 percent higher than post-synthesis, and timing still could not be met. The team then ran the critical block RTL through Encounter RTL Compiler global synthesis. The resulting netlist also met their pre-layout timing and power goals. The physical synthesis of that netlist completed in 10 hours, meeting timing without relying on too many leaky low-voltage threshold cells and extra buffers. *See Table 3*.

Metric	Old synthesis	Global synthesis
Synthesis pre-layout timing	Met	Met
Synthesis pre-layout power	366 µW	369 µW
Physical synthesis timing	Failed with –126ps violation	Met
Physical synthesis leakage power	553 µW	401 µW

Table 3: Design-project study 2—timing and leakage power results

A third design-project case study focused on maintaining the die area savings that came from cell area reduction Encounter RTL Compiler global synthesis. This 450K-instance chiplet test case came from a very high volume graphics design. The pre-layout cell area savings from using globally based synthesis algorithms was 9 percent. Then, both netlists were run through place and route with the same utilization factor (i.e., the bounding box for the globally based netlist was set 9 percent smaller than the one for the other netlist). Both designs met timing in this case. However, the globally-optimized netlist required 22 percent fewer buffers through the back-end timing convergence process. This resulted in further area savings that provided an additional buffer of a 1 percent decrease in utilization that was very critical in the design, along with the 9 percent area savings.

CONCLUSION

Renewed focus on the front-end of the design process, as well as the global logic structure, can provide maximum leverage for design closure for future high-complexity projects. The impact of synthesis on design closure already has proven to be dramatic. Applying a globally based set of optimization algorithms decreases the engineering effort required to achieve acceptable results. This innovative global approach can surpass what was possible using previous methodologies that were based on local synthesis techniques.

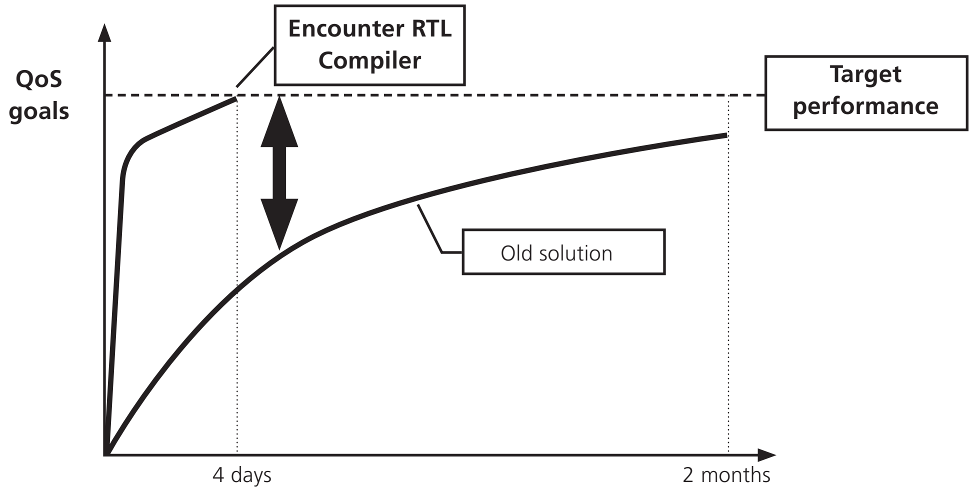


Figure 8: Results and speed of closure

The new generation of globally based algorithms measurably reduces the effort required to meet design goals throughout the remainder of the implementation process (*see Figure 8*). By applying these algorithms to optimize synthesis, high-quality results and design closure can be achieved more quickly.

For more information about
this and other products contact:

info@cadence.com

or log on to:

www.cadence.com

cādence™

Cadence Design Systems, Inc.

CORPORATE HEADQUARTERS
2655 Seely Avenue
San Jose, CA 95134
P: +1.800.746.6223 *(within US)*
+1.408.943.1234 *(outside US)*
F: +1.408.943.5001
www.cadence.com