Maximizing Leakage Savings with DC 2010.03

March 2010

Mary Ann White
Implementation Group
Synopsys

Design Compiler® 2010.03 boosts designer’s productivity with many new advanced features. The release includes many new optimization technologies delivering smaller area, faster timing, lower dynamic power and lower leakage power.

This paper highlights how use of concurrent optimization for multiple modes and at multiple corners (MCMM) with Design Compiler Graphical provides users with maximum productivity and post-route leakage savings.

Multi-Corner, Multi-Mode (MCMM) Optimization

Today’s complex designs need to operate under multiple conditions and in many modes such as scan, sleep and other functional modes. Optimizing serially across each mode and several operating conditions (corners) can be time-consuming and require multiple iterations to achieve optimal results. With Design Compiler Graphical, RTL designers can analyze and optimize designs across multiple modes and corners concurrently to provide additional leakage savings and a better starting point for place and route products, such as IC Compiler.

A corner is defined as a set of libraries characterized for process, voltage and temperature variations. Corners are not dependent on functional settings; they are meant to capture variations in the manufacturing process, along with expected variations in the voltage and temperature of the environment in which the design will operate. A mode is defined by a unique set of clocks, supply voltages, and timing constraints in similar operating conditions. It can also have annotation data, such as SDF or parasitic files.

MCMM optimization is useful for designs that can operate in many modes such as test mode, low-power active mode, stand-by mode and so on. Used along with specification of power intent in the Unified Power Format (UPF), it serves as the key enabling technology for performing dynamic voltage and frequency scaling (DVFS) design realization.

Implementation of a design must take into account all of the different operational modes that a design can have. For example, a single block can operate in fully functional, low-power active, stand-by and/or completely shutdown modes. Without multi-mode optimization, the typical flow is to perform timing optimization sequentially for the different modes targeting various corners that represent the different operating conditions and constraints. Design Compiler Graphical’s multi-mode concurrent optimization reduces the number of iterations providing faster time to results for multi-mode designs.
One of the primary benefits of performing multi-corner optimization is to get optimal leakage results. Prior to the availability of Design Compiler Graphical MCMM optimization, users typically used one of these two techniques for optimization of their designs:

a. Perform leakage power optimization on the same corner as timing optimization
b. Perform leakage and timing optimization sequentially using different corners for worst case timing and worst case leakage

The graph below (Figure 1) shows the effects of the worst case timing and leakage corner of a 65nm low power process where the higher voltage corner (1.32V) reflects the best performance but leakage is at its worst. Conversely, the low voltage corner (0.9V) reflects the best leakage; however, performance (or speed) of the design will be at its slowest at this environmental condition. Both of these corners must be accounted for simultaneously in order to achieve the optimal leakage and performance objectives for the design.

In addition, with the complexities of sub-micron processes, the worst case leakage corner is now changing from the typical hot temperature (125C) at VDD+10% case to a temperature inversion corner usually occurring at a colder temperature environment. Design Compiler Graphical’s MCMM optimization can take all of these different process corners into account to provide the best leakage results while trying to minimize impact to performance.

Design Compiler Graphical uses scenarios to perform MCMM optimization. A scenario is a mode or a corner or a combination of both that can be analyzed and optimized. MCMM Optimization involves managing the various scenarios defined for the design. Design Compiler Graphical automatically prunes the set of active scenarios provided to a set of dominant scenarios. A dominant scenario is one that has the worst slack among all the scenarios for at least one of its constrained objects (delay constraints associated with a pin, DRC constraints for a net, leakage power, etc.). Design Compiler Graphical performs optimization across all of the dominant scenarios it identifies and provides a single report that shows the leakage and timing performance for all of the different scenarios.

Figure 1: Worst case leakage and timing corners
MCMM performed during the front-end synthesis process can effectively be used to achieve optimum leakage results post-back-end. Actual leakage savings may vary based on design objectives, but performing MCMM optimization in both Design Compiler Graphical and IC Compiler results in an average of 9% improvement in leakage savings as opposed to performing MCMM optimization in IC Compiler alone. The graph below (Figure 2) shows the leakage improvement obtained by running MCMM optimizations in both Design Compiler Graphical and IC Compiler.

![Figure 2: MCMM results](image)

The 2010.03 release allows users to specify a goal of achieving the lowest leakage possible regardless of performance targets. Used in conjunction with new infrastructure changes, the 2010.03 release of Design Compiler Graphical introduces support for leakage only corners improving MCMM runtime an average of 25% over the 2009.06 release. With leakage only scenarios, timing is not taken into account resulting in faster runtimes. The graph below (Figure 3) shows runtime improvement for designs using MCMM leakage only scenario support available in the Design Compiler Graphical 2010.03 release.

![Figure 3: Runtime improvement for designs with leakage only scenario](image)
Another application where MCMM optimization can be applied is to perform DVFS optimization and analysis. UPF allows the user to specify the different modes of the design using the power state table (PST) capability to define the different states that the design can have (multi-mode). Dynamic frequency is covered via use of multiple SDC files, where various clock frequencies can be specified along with the different operating conditions. The table below (Table 1) shows a sample of what a DVFS design might look like:

<table>
<thead>
<tr>
<th>Design Blocks</th>
<th>State S0</th>
<th>State S1</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLK (specified in SDC)</td>
<td>7ns</td>
<td>10ns</td>
</tr>
<tr>
<td>Top</td>
<td>High Volt</td>
<td>High Volt</td>
</tr>
<tr>
<td>BlockA</td>
<td>Low Volt</td>
<td>Low Volt</td>
</tr>
<tr>
<td>BlockG</td>
<td>High Volt</td>
<td>Low Volt</td>
</tr>
<tr>
<td>BlockZ</td>
<td>High Volt</td>
<td>High Volt</td>
</tr>
<tr>
<td>BlockC</td>
<td>High Volt</td>
<td>High Volt</td>
</tr>
</tbody>
</table>

**Table 1: DVFS design example**

The MCMM feature also provides ease-of-use and compatibility between flows in Design Compiler and IC Compiler. Both products share a common MCMM setup and interpretation throughout the flow which is compatible for use with sign-off analysis in PrimeTime. Since the setup is the same, it is possible to achieve accurate timing and power correlation per scenario defined.

**Summary**

The 2010.03 releases of DC Ultra and Design Compiler Graphical provides the ability to concurrently optimize for multiple modes and corners reducing iterations between the front- and back-end implementation allowing designers to achieve rapid design convergence. With the 2010.03 introduction of leakage only support, productivity is enhanced via significant runtime improvements.