Power grid analysis: Ripe for change

Christen Decoin - April 29, 2014

Recent industry reports estimate the size of the power grid analysis (PGA) market at about $100M, with a double-digit year-to-year growth. Interestingly enough, it is the only EDA market of this size that is not dominated by one of the big three EDA providers, but by a former start-up, Apache Design. Apache essentially created PGA from the ground up around the turn of the century—it was the trailblazer that developed the technology and created awareness of the need for PGA. As a result, Apache’s solution was adopted as a point tool by most of the semiconductor companies, who expected that Apache would eventually be acquired by one of the big EDA providers. However, during Apache’s IPO attempt, they were acquired by ANSYS, rather than an EDA company, which was a big surprise to the semiconductor industry.

Since the acquisition, activity in the power analysis market has increased, with the emergence of a few start-ups—somewhat unique in the EDA industry, where start-ups are becoming extremely rare—and, more recently, with the release of a product update from Cadence. Because the Apache IC PGA tool is a rather isolated offering within the overall ANSYS portfolio, EDA vendors see an opportunity to provide a solution that is better aligned with existing tool flows. Most semiconductor companies would rather acquire PGA tools from their preferred EDA vendors, but more importantly, IC designers working at the most advanced nodes are experiencing capacity and performance limitations in the incumbent tools, which have architectures dating back 15 years.

At first, these limitations were only apparent in the largest IC designs (e.g., FPGAs, GPUs, large CPUs, server chips), but now designers are facing the same issues with SoCs. These limitations forced designers to take an unusual and difficult approach to PGA, known as slicing (Figure 1). They run a chip through power analysis until the analysis crashes; then they repeat the process on successively smaller and smaller slices of the chip until they finally partition the chip into slices the PGA process can digest without failing. Even then, each of these small slices usually require several days of runtime using large memory machines. Finally, after successful analyses of all the slices, the results must be merged, and then debugged, because of the errors and inaccuracies introduced by the slicing process. This methodology is not only extremely slow, both in flow setup and implementation, but it is also not scalable, since it must be repeated for each modification to the design.
Solution providers also tried addressing PGA capacity and performance issues by using hierarchical extraction, but this approach has serious limitations. Hierarchical extraction, in which the RC netlist representation of the power grid is executed hierarchically, leverages hierarchical DEF—once the power grid extraction is completed hierarchically, the analysis is performed flat at the top level on the merged netlist. The extraction of a sub-block without taking into account the top-level metallization can lead to inaccurate results. Accuracy is also an issue when PGA is performed under the assumption that blocks are independent, because in-die variation calculations are sensitive to context.

Another approach, called hierarchical PGA, consists of running power analysis on sub-blocks to create power models representing these blocks, then running the full chip analysis using the power models (black boxes) in place of the actual sub-blocks (Figure 2). Hierarchical PGA leverages hierarchical DEF to run the full PGA flow on each sub-DEF. A compact model is generated for the block, and the analysis is run flat at the top level This approach has been used successfully for GPU designs, because they have an extremely robust grid with a lot of redundancy, and have extremely regular and repetitive sub-blocks (100s of repeated “chiplets”), which is ideal for hierarchical PGA. Unfortunately, hierarchical PGA cannot be applied successfully to SoCs, CPUs and server designs—to be able to leverage such flows, the power grid mesh must be fully described in the hierarchy. In addition, there is less redundancy impacting the grid robustness, interaction between sub-blocks is less predictable, and inaccuracies can be generated due to the standalone analysis.

What users really need is the ability to do fast and accurate PGA on a flat representation of the design, no matter what the design size. So far, the available tools have been unable to provide this capability. To further complicate matters, it is apparent that designers must now consider power grid design much earlier in the design cycle. The increasing complexity of designs, the addition of multi-patterning constraints, and the introduction of FinFET constraints has made power grid design more critical than ever before. Designers can no longer wait until signoff to verify that the power grid will be adequate and reliable. This shift puts existing tools in a tough spot, because they are not easily integrated into existing physical design flows to provide efficient upstream analysis.

A host of new requirements, and the difficulties of achieving efficient power analysis with existing tools and techniques, has brought PGA to the forefront of designers’ wish lists. We expect to see renewed innovation in this segment by EDA vendors, who will deliver a new generation of tools that not only have the accuracy and capacity needed for advanced node designs, but also integrate easily into customers’ existing flows.

Also see:
- Power-grid analysis on SOC-graphics-chip design
- Power Noise Reliability Sign-off of Custom and Analog IPs
- Formal techniques solidify power-grid verification