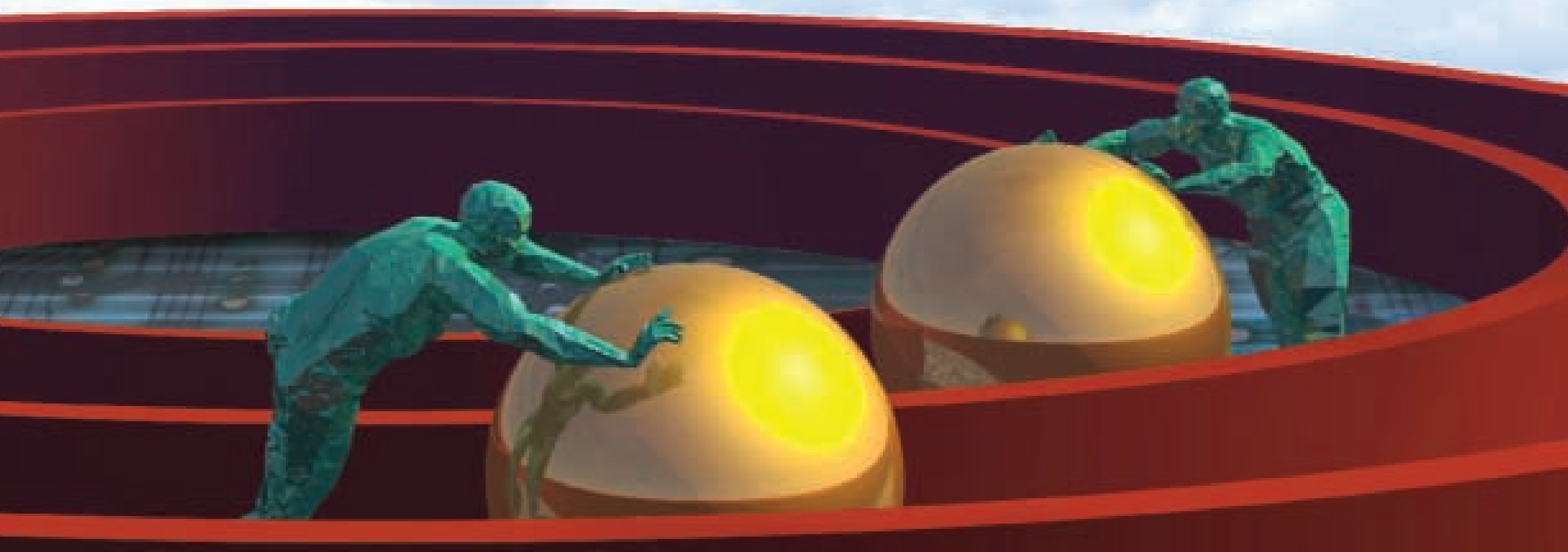
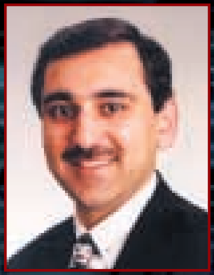
Leveraging ASIC Expertise for Platform FPGAs

As Platform FPGAs encroach into ASIC territory, the need for ever more complex synthesis tools and strategic alliances with industry-leading EDA vendors becomes essential.



by Sanjiv Kaul
Senior Vice President and General Manager
Physical Synthesis Business Unit, Synopsys Inc.
www.synopsys.com



In a relatively short period of time, programmable logic has evolved from simple glue logic to system-on-a-chip (SoC) prototypes to Platform FPGAs for complex system applications. These Platform FPGAs, as exemplified by the Xilinx® Virtex™-II family, are being targeted for applications that were once solely the domain of ASICs. Now, designers from many different industry segments – communications, medical imaging, graphics processing, and consumer electronics – work with FPGAs.

As devices and applications grow in size and complexity, designers are increasingly applying ASIC-like design and verification techniques to take advantage of the new array of capabilities of Platform FPGAs. The reasons for this vary with the designer's perspective. In some cases, a team verifying an ASIC SoC wants the Platform FPGA prototype to go through the same flow. A designer who used an ASIC on the last project might want to use familiar and proven tools. Many designers who are used to simpler FPGA tools will want to know how ASIC designers have addressed the challenges now faced by Platform FPGA designers. For all these reasons, it's important to ask what has made ASIC and standard cell designers successful in the face of burgeoning complexity and how their EDA tools and flows can apply to Platform FPGA design.

More Sophisticated Synthesis

For smaller designs, push button synthesis tools have met the needs of FPGA designers. As FPGAs get into the million-gate-plus count and support complex clocking schemes, much more sophisticated synthesis technology is required. ASIC designers are accustomed to employing much more sophisticated synthesis methodologies than the current FPGA synthesis tools provide.

More often than not, ASIC designers realize that the fastest path to silicon is to assert a great deal of control over the synthesis process to meet performance goals. Designers of complex ASICs occasionally use a top-down synthesis methodology to handle time budgeting and to give them an overall idea if the design will meet timing goals. This top-down methodology can produce quick results if the design is not pushing the device's performance limits.

Designers at the leading edge of the performance curve, however, also want a bottom-up synthesis capability to focus on the blocks where timing and/or area are tight. During block synthesis, they need precise control of the design hierarchy and a wide range of design constraints. As designers use FPGAs to implement more challenging designs, they find that only some of the features they need currently exist in today's FPGA synthesis tools. They are demanding more ASIC-like capabilities to get their jobs done.

Combining Synthesis and Placement

One of the hottest trends in standard cell design is physical synthesis, and this area is heating up for Platform FPGA design as well. True physical synthesis offers more productivity and better performance by integrating synthesis and placement into a single optimization step. By incorporating placement, the optimization algorithms get more accurate timing data for successful timing closure. Currently, FPGA designs are synthesized and placed in two distinct steps. However, even when data is passed back to synthesis after placement, it is difficult to achieve timing closure on the largest designs. A true physical synthesis solution is needed for FPGA designers to reach timing closure and realize fast time to market.

Successful FPGA and ASIC designers also strive to do two things with their HDL code: make their HDL code as technology-independent as possible, and verify functionality first – because it costs less to fix an error detected early in the design process. The latest in technology independence is power management in SoC designs. Many SoCs use clock-gating schemes in a standard cell implementation. To implement or prototype the same functionality in an FPGA requires clock enables. A complete ASIC tool flow allows the designer to write technology-independent code, which relies on power synthesis tools to insert the clock gating for the standard cell SoC. FPGA synthesis then uses that same code to target the clock enables on the FPGA without the need for any additional technology conversion steps.

Complexity Demands Robust Verification Flow

Platform FPGAs also need ASIC-like performance in the verification stages of the design. The team that verifies a design often doesn't care whether the final implementation is in standard cell or FPGA technology, they just want to prove it works. That means that the same speed and accuracy that they have come to expect from today's ASIC verification tools must be available, regardless of the final device's implementation. This verification process must also feature test bench generation to handle the soaring gate counts and complex vectors used to verify Platform FPGAs.

Many of the same verification bottlenecks brought on by deep submicron processes also apply to complex FPGAs. Verification engineers need and demand formal verification tools to avoid multiple iterations that can stall a design. Formal verification – in particular the equivalence checking method of formal verification – has recently entered the mainstream of high-capacity, complex designs. This mathematical proof of functional equivalence between two versions of a design can comprehensively verify in a matter of minutes or hours a design that would take weeks to verify using a gate-level simulator. However, to be truly effective as a replacement for gate-level simulation in today's design flows, a formal verification tool must offer more than capacity and speed. It must also provide fast and easy debugging of design errors, it must be usable in a broad array of design types and applications, and it must be easy to integrate into existing design flows.

All verification tools must also provide a feedback loop into implementation, or more valuable design time will be lost. Some of today's formal verification tools meet these requirements, but Platform FPGA designers must be sure that their entire verification environment is suited to verify complex designs.

Another way to reduce the need for gate level simulation is static timing analysis – a technique that both FPGA and ASIC designers have been using for quite some time. Static timing analysis checks the delay on all paths of a chip, providing an exhaustive check of timing to a given set of constraints. What's new for the Platform FPGA world is the need to model the timing of complex IP such as processors and ROMs. FPGA designers also need the application of advanced debugging aids to quickly locate the root cause of timing issues – and feed constraints back to earlier stages of the design process.

Conclusion

For designers to meet their goals with Platform FPGAs, EDA, and FPGA vendors must work together closely to ensure that the ASIC design success story can translate into Platform FPGA technology. Synopsys is working closely with Xilinx to ensure that designers who use Platform FPGAs and supporting EDA technology can get the same performance, reliability, and ease of use that the ASIC design community has enjoyed. As a premier EDA company and synthesis pioneer, Synopsys is well poised to apply its extensive knowledge of ASIC design tools and flows to the Platform FPGA arena. Working with Xilinx and its mutual customers, Synopsys will constantly strive to accommodate the myriad changes that will undoubtedly occur as Platform FPGAs take their place alongside ASICs for complex system design.