Electronic Design

JUERGEN JAEGER, Director of Prototyping Product Strategy,
Siemens EDA,

Is Custom Silicon Coming for FPGA Prototyping, Too?

Today's ultra-complex SoCs are pushing traditional FPGA prototyping to its limits. A new approach uses ASICs that are purpose-built for prototyping.

A growing number of advanced system-on-chip (SoC) designs are being deployed not only by the traditional merchant IC design companies but also by a more non-traditional set of companies: cloud providers, automakers, and consumer-goods manufacturers. These designs are highly advanced, targeting leading-edge manufacturing nodes, and they contain an ever-increasing software component. They're also huge, with high gate counts, and rely on complex on-chip networks, SRAM, power, clock, and test architectures.

One of the key differences lies in the purpose of the chips developed by these systems companies. Such chips are often less general-purpose and more application-specific, with engineers designing them to suit the needs of specific hardware and software system environments.

With systems companies pursuing some of the most ambitious designs in the industry, FPGA-based chip prototyping, lately also called "software prototyping," is evolving to keep up with these complex ASICs.

For application-specific chip designs of this ilk, verification means more than ensuring that the hardware meets the requirements outlined on a datasheet. It means confirming that both the chip and the software running on top of it work within the larger system. This involves traditional RTL verification and validation of the entire system, including — and this is key — the software stack and interactions between the chip, the PCB, and mechanical subsystems.

Projects of this scale require hardware, software, and system-level co-design. While verifying a modern chip is an extraordinarily complex task, verifying it in the context of a complete system is even more challenging.

The Pros and Cons of FPGA-Based Prototyping Systems for SoCs

Isolated testing isn't enough to really understand the interactions between hardware, software, and the overall system. It requires running real production workloads on the relevant portion of the design RTL model and then observing everything from the state of the RTL model to the behavior of the software and behavior of the external system interfaces. Engineers must be able to capture and analyze interesting sequences of events at full speed.

Even more important is the ability to trace an error condition back to its root cause, even if the root cause lies deep in the RTL model or in the software stack. That level of visibility is key to diagnosing and resolving the kinds of issues that arise at the intersection of the hardware, software, and system.

Traditionally, engineers addressed this challenge by building home-spun FPGA prototyping boards. When the most advanced SoCs only contained millions of gates, it was still practical to map the RTL design onto one or two FPGAs, add some virtual logic-analyzer functions, compile the model on third-party tools, and connect the FPGA board to the target system board. But as SoCs continue to grow in size and complexity, this approach becomes increasingly limited.

Depending on the expertise of the design team, building these prototyping systems can be demanding and time-consuming, particularly when it lies outside the core skills of a team. These systems also often produce results that can be unpredictable. Thus, it leaves open the question of whether a bug being observed is actually a bug in the design or simply caused by issues within the FPGA prototyping system itself.

Additionally, on larger designs, this approach required verification engineers to switch between the simulation, emulation, and the FPGA prototyping environment, often with different user interfaces and databases. That discontinuity causes the FPGA model to diverge from the chip RTL model, leading to further inconsistencies.

This approach doesn't scale up to today's challenges in terms of logic capacity or the time-to-market pressures

faced by engineers. Modern designs are too large, and the interactions between the chip, software, and system — and, consequently, between the RTL verification and the prototyping efforts — are far more complex.

For example, a modern AI accelerator can consist of thousands of compute engines and large blocks of on-chip RAM connected by a network-on-chip (NOC), supported by several large CPU cores. Critical interactions may involve a CPU, a bank of compute engines and RAMs, remote direct memory access (RDMA) controllers, high-bandwidth memory-channel controllers, external high-bandwidth memory (HBM) stacks, and an external interface processing unit.

Monitoring and debugging these complex interactions means implementing a large amount of RTL in the prototype, often demanding far more than a handful of FPGAs. Managing all of these FPGAs along with the high-bandwidth interconnects between them and external system boards can be a significant headache for the design team. The verification teams also need the flexibility to move between the hardware emulation and prototyping systems as the design evolves. Any disconnect between these environments can slow down the entire process.

Purpose-Built ASICs for More Scalable SoC Prototyping

The solution is the same one many systems companies already rely on: use a purpose-built ASIC. Today's commercial ASIC-based prototyping platforms, such as Siemens' Veloce proFPGA CS platform, are designed to overcome many of the limitations of traditional FPGA prototyping.

When evaluating a prototyping platform, it's also important to consider how well it integrates with the rest of the verification flow, including emulation and software validation environments.

For example, Siemens Veloce proFPGA CS platform has exact correlation to the Veloce Primo CS enterprise prototyping system, and the Veloce Strato CS emulation platform. As a result, design teams can quickly move from RTL verification on Strato CS to early software validation and system verification on proFPGA CS.

Another crucial aspect when selecting a FPGA-based prototype solution is scalability. Throughout the design process, a prototyping platform needs to accommodate a wide range of models.

Initially, the model may involve a small protocol controller interacting with an external communications interface, which easily fits into a single FPGA. Later stages may require studying traffic patterns between compute clusters and HBM under realistic workloads, requiring multiple FPGAs. In cases such as evaluating the emergency response of an ADAS, an almost full-chip model and complete software workload may be needed.

To address this range of model sizes, a modern FPGA-based prototyping system must provide scalability. Utilizing the same compile flow and run-time interface, a platform must scale from a single FPGA to multi-FPGA desktops, racks of FPGA-based blades, and multi-rack installations — all with single-FPGA granularity. This range employs unified interconnect architectures, allowing model-preparation tools to distribute models across chips, boards, and racks.

Conclusion

Designing leading-edge SoCs requires verifying the full-chip RTL with the production workload in real systems. Modern FPGA-based prototyping systems are essential and critical for successfully conducting the verification process. They provide detailed exploration of chip-to-system interactions while maintaining the ability to review behavior in the full-chip, full-workload model.

But to keep up the fast-paced electronics landscape, it 's becoming critical to upgrade aging generations of in-house FPGA-based prototyping systems and invest in a more purpose-built solution.