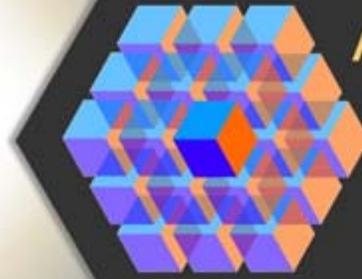


sponsored by

Gabe on EDA • EDAMarket



Assembling the Future

A Newsletter About the Design
and Production of Electronics

ISSUE 012 • FEBRUARY 2012

[SUBSCRIBE](#)

[PDF and Archives](#)

[twitter](#)

In this issue:

- [Verification Is More Important Than Ever](#)
by Gabe Moretti
- [The Next Major Shift in Verification is SoC Verification](#)
by Michael Sanie
- [The Truth about SoC Verification](#)
by Adnan Hamid
- [Chip Design Without Verification](#)
by Lauro Rizzatti
- [Verification – You Have to Start Smart](#)
by Mike Gianfagna

Verification Is More Important Than Ever

Gabe Moretti

Some years back, comparing the relevance of DAC and DVCon to the electronics industry would have been unthinkable. Of course DAC, a conference that covers, or aims to cover, every aspect of electronic design, is more important. Today, though, I can see the time, just a few years from now, when every single step after producing an RTL description of the design is in the hands of very specific tools carefully chosen and integrated under the guidance of the foundry that will manufacture the design. Verification tools, specifically those used above RTL, will instead continue to be chosen by the project team, or the corporate CAD department.

To find those tools engineers will attend DVCon. Since February is traditionally DVCon month, this issue of the newsletter is dedicated to verification. The four contributed papers all point out, from different perspectives, the complexity of verifying a design. Although tools have improved, both in

capacity and in methods, the difficulty of verifying a design has increased at a pace even greater than the increase in design size.

The reason, of course, is that size is not the only agent in complexity increase. The architectural sophistication of today's designs is greater by many orders of magnitude than that of designs implemented twenty years ago, as the article by Michael Sanie of [Synopsys](#) shows.

Adnan Hamid, CEO of [Breker Verification Systems](#) cleverly uses a iceberg to point out the hidden challenges to verification engineers and consequently to designers. Generally no one makes intentional mistakes, so it is the less obvious, the unexpected execution states, the unprecedented use mode, that create problems in the product when they are not isolated by a good verification methodology and flow.

Lauro Rizzati of [EVE](#) points out the danger of relying solely on one's experience. The complexities of verification are such that software simulation is almost never enough. Emulators and hardware accelerators are required to verify that complex software, like operating systems, executes properly with the hardware system. Technical advances in both FPGA and hardware systems design have lowered the cost of emulation to the point that practically every project can afford to use this method.

One of the most important development in verification is the creation of usable formal verification tools. For a few years, formal verification seemed to be a mathematical method usable only by few highly trained individual. But this has all changed now and "mere mortals" like IC designers and verification engineers, can avail themselves of powerful formal methods that both increase the effectiveness and decrease the execution time of verification suites. Mike Gianfagna of [Atrenta](#) writes about this in his Viewpoint.

More needs to be done, of course, and EDA companies are working hard to develop more efficient verification tools and methods. We need to find a way to increase the difficulty of making errors. Less errors means less verification costs, but correct by construction is still just a goal. Hardware subsystems integration as well as hardware/software co-development and integration are promising areas where the design and verification methods can converge to provide a more robust development environment. And all this bodes well for DVCon, a place devoted to the integration of design and verification sciences into a productive whole.

The Next Major Shift in Verification is SoC Verification

Michael Sanie, director of verification product marketing, Synopsys, Inc.

We have witnessed two major shifts in verification during the past two decades. During both, design

teams have been able to rise to the challenge of verifying the most complex state-of-the-art designs by applying innovative verification technology.

The first major shift took place in the 1990s. At that time, a state-of-the-art ASIC consisted of about 5 million gates in 1µm-0.5 µm process technology. The computing industry was driving the complexity curve in its quest to produce more powerful CPUs (Figure 1).



[Figure 1: Major shifts in verification needs – driven by design complexity]

During the 1990s, design teams were transitioning from gate-level design to the use of hardware description languages (HDLs). This enabled design teams to tackle even more complex designs. The main method for verification at this time was HDL simulation. Using HDLs, design teams were able to design larger and more complex chips efficiently, and scale their design efforts. But the efficiency and scalability benefits that HDLs brought to the design process were not extended to the simulation efforts. The gap between design and verification scalability continued to grow because HDL simulation was not able to keep up with the rise in design complexity and size. The “simulation productivity gap” emerged as the most pressing challenge in verification.

The emergence of native compiled-code simulation technology helped verification teams to significantly improve their simulation productivity. Synopsys VCS®, as the industry’s first compiled-code simulation technology, revolutionized simulation and addressed the simulation productivity gap.

A comparable shift also occurred in the 2000s. In the early 2000s, design complexity, now being driven mostly with networking applications, reached higher levels. Design teams used more and more IP as ASICs became increasingly complex, reaching gate counts of 10 million or more.

Verification teams looked beyond simulation and started to turn to more sophisticated technologies, including the use of advanced testbenches, constrained random approaches and assertions as they worked towards achieving higher levels of verification coverage. While these new verification technologies existed as point tools, verification teams had to put in significant effort to make them work together, and it became increasingly difficult to create scalable verification solutions to address their most complex designs. As the designs’ sizes and complexities continued to grow, the

environments needed to manage the verification of these designs were once again no longer scalable and efficient.

Synopsys works with industry leaders to address this new “verification productivity gap” by introducing SystemVerilog and advanced testbench methodologies. In addition, introducing native testbench technology enabled design teams to combine and integrate several verification approaches around SystemVerilog. Together, these major innovations helped verification teams to scale their verification solutions once again to be able to handle more complex designs and address the verification productivity gap. Going forward, Synopsys and other industry leaders worked together to first drive SystemVerilog as the ratified industry standard for design and verification, and then UVM as the ratified industry standard for SystemVerilog-based verification methodology.

Fast Forward to Today

Several years after the emergence of SystemVerilog and advanced verification methodologies, we are now witnessing another substantial shift in the industry’s design and verification needs. Today’s requirements are being driven by changes in the SoC design process and the continued growth in complexity.

Sub-32 nanometer (nm), 100 million+ gate designs characterize today’s SoC devices. In order to meet project schedules, design teams are making extensive use of IP. Today’s convergent consumer products demand SoCs which integrate applications processors along with several other functions and extensive support for software applications.

Factors	Early 2000	Today
Gate Count 	10M+ Gates	500M+ Gates
Voltage Domains 	1 Voltage Domain	20+ Voltage Domains
Cores 	Single Core CPU	16 Core CPUs
Protocols 	1 or 2 Protocols	10 to 15 Protocols
Software Content 	4% SoC Value is SW	25% SoC Value is SW

[[Figure 2: Increasing SoC complexity](#)]

Market Trends Drive Technology

SoCs today look very different from those of the early 2000s. The complexity of design specifications has changed significantly. They are much faster, incorporate more features and functionality, and support extended runtimes on the same battery technology.

Convergence is the key market trend for today's devices. Convergence products, such as smart phones, tablets, and other advanced consumer products, combine several key technologies within the same device. They typically incorporate multicore CPUs supporting multiple interface protocols – upwards of 10. They require long battery life, advanced software features, and short time-to-market. Design complexity is increasing because of the need to achieve low power and some designs now incorporate more than 20 voltage domains.

Time-to-market is a critical business issue. Consumer markets make choices faster than ever before and a delay of just a few weeks can make the difference between product success and failure. We only have to look back at the recent emergence of the tablet sector for evidence of how being second to market can have a devastating effect on initial product success.

For many consumer products, software is now the key to SoC differentiation. It should be no surprise, then, that according to IBS Research, 25% of the value of today's SoCs is in the software. This is up from just 4% in the early 2000s.

It now takes significant investment to develop an advanced SoC. Companies are spending upwards of \$100 million to produce the latest chip designs, with the majority of that spend directed at infrastructure and engineering costs. The size of the verification team and effort typically outweighs that of the design by two-to-one. It is no surprise that businesses are focusing on verification productivity like never before.

Design Complexity Impacts Verification Need

The effect that these market trends have had on verification is profound. Through our collaboration with leading design companies, we are fortunate to work on many leading-edge designs. Over 60% of designs on processes of 45nm and below, and more than 90% of designs of 32nm and more advanced nodes are verified with VCS. This exposure to advanced designs gives us a unique insight into the “verification profile” of a state-of-the-art design today. The metrics, some of which are as high as the figures below, are staggering:

- Tens of millions of lines of RTL and testbench code
- Larger designs could need hardware with over 150GB RAM for verification
- Multiple (10 or more) protocols on a single chip
- Hundreds of thousands of assertions
- Tens or hundreds of power domains
- Over a terabyte of coverage data to analyze

In an attempt to keep up with large verification requirements, compute farms have doubled in size over recent years, verification teams have become twice as large as their design counterparts, and the debug process now accounts for 35% of the entire verification effort.

Once again, the industry needs major advancements in verification to accommodate the dramatic shift in the design landscape.

The Next Major Shift in Verification

Verification teams do embrace enhanced feature sets, improved simulation performance and more

efficient use of memory offered by verification providers. Yet, incremental improvements to today's tools will not be sufficient to deliver an order of magnitude boost to verification productivity given the complexity of today's designs. Instead, verification teams need innovations in verification.

It is apparent that what verification teams need today – if they are to successfully get beyond the challenges outlined above – are innovations that focus on key productivity bottlenecks. These innovations need to deliver significant improvements in performance and capacity; superior, more intuitive debug that enable engineers to quickly analyze vast amounts of data and find design bugs; comprehensive, proven verification IP that is fast, efficient, and timely; innovative low-power verification solutions; and hardware-software co-verification solutions that allow software teams to develop code alongside the hardware and validate the entire system functionality and performance.

Solving these difficult problems is a significant growth opportunity. Synopsys continues its drive to innovate and invest in verification. Synopsys' innovation was at the heart of the first two major shifts in verification.

This article was first published in the Synopsys Insight newsletter:

<http://www.synopsys.com/Company/Publications/SynopsysInsight/Pages/default.aspx>

About the Author

Michael Sanie is director of verification product marketing at Synopsys. He has more than 20 years of experience in semiconductor design and design software. Before Synopsys, Michael held executive and senior marketing positions at Calypto, Cadence and Numerical Technologies. He started his career as a design engineer at VLSI Technology and holds four patents in design software. He holds BSCEE and MSEE degrees from Purdue University and an MBA from Santa Clara University.

The Truth about SoC Verification

Adnan Hamid, Chief Executive Officer, Breker Verification Systems

Functional verification of large, complex chips is hard so plenty of technology has been developed to enable the use of both formal analysis and testbench-based simulation on such chips. Verification remains a difficult task that consumes more than its share of project resources, but many large, complex chips are being taped out successfully today.

A system-on-chip (SoC) device containing one or more embedded processors is also large and complex, but it does not follow that the same verification techniques that work for other types of chips will suffice for the SoC. The presence of embedded processors presents a fundamentally different verification challenge; the chip can no longer be verified with just a testbench manipulating inputs and outputs.

The truth is this: the SoC cannot be fully verified without running thorough test cases on its embedded processors. Functional operation of the SoC requires production code, such as RTOS, drivers, and applications, running in the processors, so it should not be a surprise that the processors play an essential role in verification as well. However, even SoC teams who understand this often do an inadequate job of verification and incur a high risk of non-functional silicon.

Most SoC verification teams follow a simple three-part process at the full-chip level:

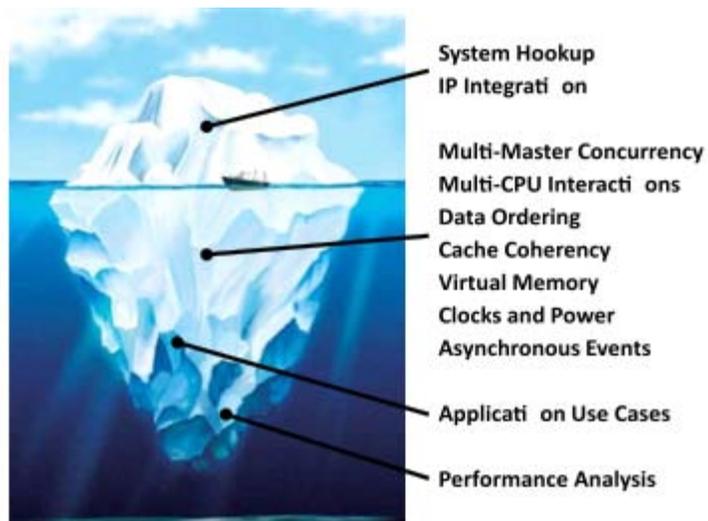
1. Verify the top-level SoC connectivity either formally or (incompletely) in simulation
2. Create a simple full-SoC testbench capable of performing some representative tests but not able to exercise a wide range of pseudo-random behavior
3. Hand-write a few simple test cases for the SoC's embedded processors to run in simulation

Verification teams relying on this "stitch-and-ship" method are implicitly or explicitly assuming that, since individual IP blocks have been well verified, only minimal verification is needed at the top level.

Nothing could be further from the truth. As shown in Figure 1, these teams are only verifying the "tip of the iceberg," while missing the most complex and error-prone functionality in the SoC.

Simple tests can determine whether IP blocks are integrated properly, but not whether they interact properly under stress conditions, such as bus contention, access to shared resources, interrupts and changes in power modes. Decisions on some of these parameters are made at the application level, so it is also essential to exercise different use cases available to the applications. Further, performance verification is critical since a performance deficiency can kill a product just as surely as a functional bug.

Although running SoC embedded-processor production code in simulation is almost always too slow, teams with access to acceleration, emulation or prototype platforms often are able to run production code for hardware-software co-verification. While valuable in its own right, production code provides sparse exercise of corner cases and does not eliminate the need for embedded-processor test cases.



[Figure : The SoC Verification Iceberg]

Some experienced SoC verification teams recognize the gap between their simple top-level testing and running production code, but they haven't known how to address it. Handwriting embedded-processor test cases is difficult, especially when threads from multiple parallel processors must interact in order to stress the corner-case conditions.

Automated test-case generation is the only viable solution, leading to a more refined version of the truth about SoC verification: The SoC cannot be fully verified without running thorough automated self-verifying test cases on its embedded processors.

The notion of self-verification is important. SoC test cases have access to the SoC inputs and outputs so they verify the chip independently, with no user intervention or interpretation unless a bug is discovered and must be diagnosed. The test cases provide all code, stimulus and checks needed for efficient verification of the SoC across a wide range of behavior and interaction among its IP blocks.

Of course, the process of generating automated tests requires input from the design and verification team on how the SoC is intended to operate. The most natural and complete way to provide this information is via scenario models, a description of behavior that starts with the desired outcome and then specifies conditions necessary to create that outcome.

"Beginning with the end in mind" is a natural way to describe behavior, leveraging the same knowledge that a verification engineer would use to hand-write embedded-processor tests. A scenario model is a more general solution than manual tests since it supports randomization, enables automated test-case generation, and is flexible enough to span from the operation of individual IP blocks up through applications and performance verification.

The use of scenario models eliminates the need for hand-written tests and fills the current gap between checking top-level connectivity and running SoC production code in hardware. Scenario models can be visualized in a graph format that clearly shows different modes of operation for the SoC and the many paths of behavior that must be exercised for effective functional verification.

It is simply unacceptable for SoC teams to continue to tape out chips with subtle bugs or performance deficiencies that arise under stress conditions never exercised in verification. Using scenario models to automatically generate thorough test cases for embedded processors brings real-world conditions to pre-silicon verification, saving precious project time while yielding a better SoC.

About Adnan Hamid

Adnan Hamid is co-founder and CEO of Breker Verification Systems. Prior to starting Breker in 2003, he worked at AMD as department manager of the System Logic Division. Previously, he served as a member of the consulting staff at AMD and Cadence Design Systems. Hamid graduated from Princeton University with Bachelor of Science degrees in Electrical Engineering and Computer Science and holds an MBA from the McCombs School of Business at The University of Texas.

Chip Design Without Verification

Lauro Rizzatti, General Manager of EVE-USA, EVE

With DVCon starting later this month, let's stop to consider a chip designed without verification. In today's world of big chips and enormous complexity, it would be an absolute impossibility. Can you imagine a microelectronic device being taped out without functional verification tools?

In the earliest days of integrated circuits, that's exactly how it happened. Circuits were designed by hand and manually laid out. As design processes became automated and moved to higher levels of abstraction, the functional complexity of the circuits outgrew the manual verification capacity of the designers. This increasing complexity, plus the corresponding rise in non-recurring engineering (NRE) costs required new verification solutions.

Enter the golden age of simulation. For multiple decades, logic simulation has been the stalwart backbone of functional verification, and it wasn't all that long ago when it was the only verification category on the pre-tapeout checklist. Simulation was more than enough.

History has a habit of repeating itself, though, and once again we are pushing the envelope of existing verification capacity. Camcorders, digital cameras, tablets — almost all consumer electronic devices — are starting to exceed the 100-million ASIC-gate threshold. These modern system-on-chip (SoC) designs often contain multiple processors and peripherals, putting compounding pressures on simulation. Large gate-counts push the boundaries of simulation capacity and degrade its performance, but the increased state-space and functionality — not to mention the increased amounts of embedded software — in these SoCs increases the number of cycles that need to be simulated by multiple orders of magnitude. The development of these big, complex designs would not be possible without the adoption of new forms of verification technologies, languages and methodologies.

Many of these new verification technologies have developed within simulation. Protocol and specification checking was introduced with assertion technology. Code coverage technology identified potential holes in the verification process. Constrained-random testing introduced through languages like e, OpenVera and SystemVerilog brought verification up to the transaction and system levels. However, these technologies and methodologies, while enhancing the verification capabilities of simulation, also place greater processing requirements onto the simulator. Simulator performance and capacity continue to improve thanks to EDA vendors such as Synopsys and Cadence, but simulation's scalability seems to be reaching its limits.

In the past, electronics companies and their design teams improved their simulation throughput by adding more PCs and tens of thousands more simulation licenses. Everything was copasetic again. But, as these companies have learned, increased throughput doesn't solve everything. Stress testing, deep corner-cases, and real-world scenarios may all require lengthy simulations that cannot

be broken down into smaller pieces for parallelization. Ten-thousand simulations running in parallel won't speed up the completion of a single test that boots Android 2.3 on a mobile-phone SoC.

Emulation, once only a small niche of the verification space, has now become a mainstay on pre-tapeout checklists, complementing the venerable logic simulator. In the past, emulators were reputed to be expensive and difficult to use, relegating their use to only the most difficult designs at companies with the biggest budgets. Emulators today are based on FPGAs and are much more cost effective, making them accessible to design teams of all sizes. As adoption has grown and as the industry has matured, emulators have also become easier to use, offering push-button compilations and simulator-like debug capabilities like waveform generation and assertion checking. The time-to-emulation has never been lower.

In today's aggressive time-to-market environment, emulation is the only way to fully verify a design's hardware and software together prior to tapeout. Virtual prototyping software from companies such as Synopsys and Carbon Design Systems has enabled software teams to start developing software earlier in the project. But without emulation, these teams may have to wait at least four months after tapeout for silicon samples. Only then can they validate and co-verify their software on the SoC hardware. As so many design teams have discovered, hardware/software co-verification issues found at this stage may trigger a costly respin and an even costlier delay in a product release.

If you're thinking that this delay is absolutely crazy, you're right. In today's SoC environment, chip design without emulation is chip design without verification.

Verification – You Have to Start Smart

Mike Gianfagna, Vice President of Marketing, Atrenta Inc.

As DVCon approaches, our world is buzzing with new verification strategies. Faster, smarter, hardware assisted – you name it, someone has a better way. The challenge of design verification is clearly a major hurdle for any complex SoC. Most industry surveys will peg this task as the major consumer of time, and thus the largest contributor to being late to market. With all this discussion and focus on verification, many approaches still miss a primary factor for success – where you start.

A remarkable amount of verification effort is spent seeking out and correcting design bugs, bugs that didn't have to be there in the first place. An alarming number of verification issues revolve around hookup errors. Another class of problems deals with clock synchronization bugs. These issues are particularly difficult to isolate, as they may exhibit a problem only every 10,000 clock cycles or so. Still more challenges are created when synthesized netlists don't match the original intent of the RTL.

All of these problems are avoidable at the start. A combination of static checks, formal methods and

early synthesis can catch these and many other design verification issues before the expensive and time-consuming process of simulation begins. Static analysis can find hookup errors well before subtle simulation problems begin to crop up. Formal methods can prove clock synchronization schemes are correct, without the need for a single simulation run. And early syntactic analysis, supplemented with fast synthesis, can catch constructs that will result in simulation/synthesis mismatches.

The leverage implied in this approach to verification can be quite large. Formal proof for clock synchronization schemes can run on the order of a few hours. Simulation runs that try to cover all mission modes for a chip can run for days. Similar comparisons can be made for the other regimes of verification discussed here.

Will this approach actually reduce the amount of time spent on verification? Interestingly, the answer is probably "no". All available time up to the tapeout will be consumed in some form of verification. Here is the interesting part, however. What if you could get past functional verification sooner? Could you then spend more time verifying the chip's performance in the context of the system it plugs into? As a lot of respins are triggered by chips that pass functional test but fail when plugged into the system, spending time on this task could truly make a difference in the spectrum of tasks we call verification. And getting it right the first time is what it's all about.

