Application Brief

ANSYS SeaScape — A Big Data Approach to Complex Chip Design

Semiconductor product design has never been more complex than it is today. Today's smaller, faster system-on-chip (SoC) designs require that the interdependent chip, package, and board be designed together, but current simulation and testing techniques based on monolithic database architectures running on large, expensive multi-core computers cannot process the massive amount of data involved. The SeaScape architecture enables engineering teams to leverage big data analytics to handle the data demands of multiphysics chip-package-board simulation and testing.

The coupling between chip, package and system design is driving a perfect storm for effective semiconductor and systems design. On one side strong coupling requires better understanding of interdependencies between design domains, and the volumes of data which must be considered in that analysis reach into terabytes (simulation, timing, layout, power, etc.). On the other side Moore's law is slowing down, which limits our ability to accelerate yet more complex analysis, even with multi-threading.

There are multiple areas in design where this is starting to create real problems. For example, we know timing closure and voltage (IR) drop are interrelated because large IR drops can cause apparently safe paths to fail. But estimating an acceptable IR drop per instance is not so easy. Conventional methods based on worst use-case margins are likely to be overly pessimistic since they don't allow for functional correlation between connected gates in practical use-cases. IR drop fixes result in overcompensation in supply rails and/or adaptive voltage scaling (AVS) controllers. Die area and therefore unit cost will be higher than it needs to be.

Now think about the impact on routability of large via arrays between the power grid (PG) mesh and the PG rails. Implementation tools tend to use a uniform user-supplied array, which blocks a number of routing layers in both X and Y directions, leading to at least some area bloat. But current demand across the die is not uniform in practical use-cases, so it ought to be possible to reduce the size of some of these arrays, reducing routing overhead and therefore die-size. In practice, product schedules limit you to using arrays tuned to worst-case current demands across the die; as a result, unit cost will be higher.

Next, consider interdependencies between die, package and system design. Power noise, which might have been more effectively mitigated on-chip, may demand additional decoupling capacitors (decaps), adding to unit cost. Reliability can be affected by current density demands; electromigration (EM) in solder joints in the package and thermal warping on the board can lead to cracked solder joints. To optimize cost and reliability, power-noise and current density objectives should be co-designed between die, package and system.

Yet another example centers on risk management. If your SoC is going into a car or a medical device, you know that standards already set high requirements for on-chip reliability, and those requirements are going to continue to rise as expectations evolve. But further increasing EM margins fails to adequately address the need for greater reliability this need because it likely increases power and area, making the device less competitive. Even then you cannot be certain that those expanded margins really cover all likely use cases. In spite of added cost, the device may not be significantly more reliable.

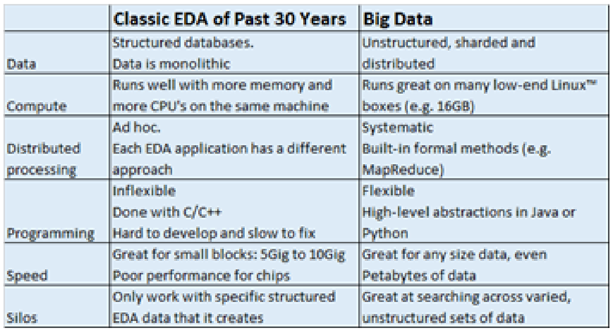
These problems could be fixed with overdesign, but that is becoming increasingly expensive. Or they could be fixed with engineering change orders (ECOs), but that only works when you only have a few problems to fix. In advanced technologies with complex designs such methods are already running out of steam.

The common theme through these examples is the need for co-analysis and co-design. You need to do power-aware timing closure. You need to do congestion-aware PG design. You need to do chip, package and system co-design. You need to guarantee use-case-aware reliability. And you're not going to get there by analyzing one domain at a time with margins to bound the other domains or by delaying problems for ECO fix.

Big Data and EDA

This and many other examples point to a systemic problem in the way EDA products are architected today. The design flow is built around monolithic tools, each of which is highly optimized to produce the most accurate results within its domain on the largest-sized piece of the design it can handle. To incorporate other forms of analysis from multiple domains into those same monolithic platforms would be wildly unrealistic; performance would become unusably slow and memory requirements would exceed all reasonable expectations.

ANSYS SeaScape is the new Elastic Compute technology architecture, modeled after the same big data architectures used in today's internet operations, but purpose-built for EDA. It allows large amounts of data to be efficiently processed across thousands of cores and machines, delivering the ability to scale linearly in capacity and performance.



	Classic EDA of Past 30 Years	Big Data
Data	Structured databases. Data is monolithic	Unstructured, sharded and distributed
Compute	Runs well with more memory and more CPU's on the same machine	Runs great on many low-end Linux™ boxes (e.g. 16GB)
Distributed processing	Ad hoc. Each EDA application has a different approach	Systematic Built-in formal methods (e.g. MapReduce)
Programming	Inflexible Done with C/C++ Hard to develop and slow to fix	Flexible High-level abstractions in Java or Python
Speed	Great for small blocks: 5Gig to 10Gig Poor performance for chips	Great for any size data, even Petabytes of data
Silos	Only work with specific structured EDA data that it creates	Great at searching across varied, unstructured sets of data

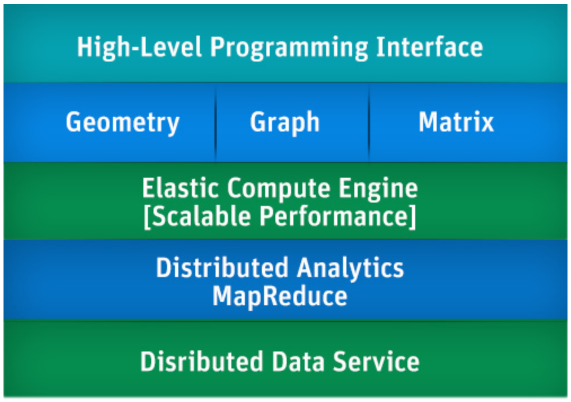
Comparison of Big data with the status quo for EDA

It is telling that systems engineering for mechanical and fluidic systems has already embraced this direction in which multiphysics analyses have become routine. An aircraft jet-engine designed one dimension at a time, using margins to bracket use cases for other analyses, would be hopelessly uncompetitive. Instead engine design looks simultaneously at airflow, thermal maps, stress maps and other characteristics to fully optimize for performance and safety, producing better, cheaper, lighter and more reliable designs.

But — and this is important — they don't try to jam all of these dissimilar and very large analyses into one tool. Multiphysics approaches couple detailed data from multiple tools together. EDA flows look quite primitive by comparison. Big designs at advanced nodes have similar needs, but we're still optimizing one domain at a time, using simple margins to bracket use cases for other domains. EDA needs to catch up with state-of-the-art approaches in other disciplines and the logical way to do that is to leverage all the data available from multiple domains — simulation, timing, layout, power, package design and board constraints — using big data analytics. SeaScape has been designed to do just that.

Actionable Analytics and the SeaScape Architecture

The primary purpose of big data analytics is not simply to have access to huge databases of different kinds of data. It is to enable decisions based on that data which are relevant to the task at hand and to do so in a short enough time that you can take action to adjust choices while the design is evolving. The SeaScape architecture is designed to do exactly this.


High-Level Programming Interface
Geometry
Graph
Matrix
Elastic Compute Engine [Scalable Performance]
Distributed Analytics MapReduce
Disributed Data Service

Big data against margin-based designs

The SeaScape architecture rests on a distributed data/file service since data (for example, simulation run data) may be scattered around many locations. On top of this sits a distributed data analytics layer based on the MapReduce concept, fundamental to all big data analytics. This conceptually splits the data (mapping) into small chunks called shards, and farms each shard for analysis. Processing can be distributed to servers as they become available, and across as many servers as needed.

There are some differences from conventional big data architectures. The MapReduce step has been enhanced to handle some of the special needs of EDA, particularly to work well with existing LSF Linux clusters. And it can flexibly handle chunks of data all the way between shards and monolithic databases. ANSYS calls all of this *elastic compute* — the ability to leverage large server environments while adjusting to the demands of availability, source data structure and optimized processing. For example, moving the compute to the data, not the data to the computer.

SeaScape also provides standard engines to fully distribute the analysis encountered in most EDA applications – geometric (for layout), graph (for simulation and timing) and matrix (for circuit simulation). Over those engines sits a Python application programming layer, used to drive whatever custom operations are needed for the steering application.

Summary

ANSYS SeaScape redefines the fundamental architecture of EDA. It enables design teams to apply modern concepts of big data analytics and scalable server computation based on standard computers for chip-package-system design convergence. The products created on the SeaScape architecture will fundamentally transform how simulation is performed.

ANSYS, Inc.
Southpointe
2600 ANSYS Drive
Canonsburg, PA 15317
U.S.A.
724.746.3304
ansysinfo@ansys.com