

Data Management: Is There Such a Thing as an Optimized Unified Coverage Database?

by Darron May, Manager of Verification Analysis Solutions and Gabriel Chidolue, Verification Technologist, Mentor Graphics Corporation

INTRODUCTION

With the sheer volumes of data that are produced from today's verification environments there is a real need for solutions that deliver both the highest capacities along with the performance to enable the data to be accessed and analyzed in a timely manner. There is no one single coverage metric that can be used to measure functional verification completeness and today's complex systems demand multiple verification methods. This means there is a requirement not only to unify different coverage metrics' but also to unify data from multiple tools and verification engines. Data management forms the foundation of any verification environment.

DATA STORAGE REQUIREMENTS

The reality of multiple tools, engines and metrics means the ideal verification database has to support more than just coverage. It has to have the capabilities to answer many questions posed by not only the verification engineer, but also the design engineer, the project manager and all other stakeholders in the verification process. The database infrastructure must provide the visibility into the process across many dimensions. The major requirements of such a database are as follows

Unification. No one coverage metric or a single verification engine can measure completeness. The database has to allow the storage of a large mix of coverage metrics from many data sources including simulation, emulation, FPGA prototyping, static formal analysis tools, software-driven tests and many other application-specific sources. It should be possible to combine data based on blocks, systems, instances, tests, users and time to give the most flexibility. Combining this data based on so many variables requires a flexible architecture and the need to store details about how, where and when the coverage data was generated. This allows the verification engineer to determine how and when a particular metric was or wasn't hit. The process also needs the ability to allow these metrics and measurements of certain system requirements

to be associated with a verification plan and ultimately the design specification.

Capacity & Performance. Unifying the verification data storage from all tools and metrics can result in huge volumes of data. The storage capacity must be able to handle the very largest of today's designs and the designs of the future. As the stored data increases it is important to have an environment that is optimized for capacity and has the performance to manipulate and query potentially large amounts of data within workable limits. Combining results from tests that have many millions of coverage bins would require such a database. This often has a negative impact on the databases' capacity and can become a tradeoff. Ideally a solution should have the ability to solve both the capacity and performance issues within the largest of projects now and in the future.

Visibility & Analysis. Allowing queries on stored verification data requires access to the database. The results from many verification engine runs need to be combined and the verification engineer needs to be able to analyze which runs with which particular settings caused particular metrics to be hit. This type of analysis is required to figure out redundancy in tests or to isolate a particular test or set of tests of a particular feature, thus allowing the verification process to be further optimized. With the combining or merging of data it's also necessary for the Verification Engineer to be able to query the database to find out information on the history of how the data was generated. This includes not only the command line options for generating the single tool runs but also the utilities used to add and combine data to the database across the progression of the project. Reductions and optimizations are required on the data so that trends can be seen across the duration of the process. The verification process is dynamic. With the addition of new functionality in the design, as well as the process of finding and fixing bugs, there is a need to be able to look at the trends of different metrics at a higher level to determine if progress is being made towards completion.

Control. With the continual data analysis throughout the project the Verification Engineer also needs to have control over the coverage model and the ability to document decisions made during the process. The database has to have the ability to manipulate the overall coverage metrics into an overall metric showing the level of completion. It also needs the ability to trade-off one metric from another based on its importance with a user controlled weighting system. As verification progresses it's also important to document any exclusions to the coverage model and the reasons why they have been excluded. These types of exclusions could be made automatically by the verification tools. An example is a static formal tool excluding unreachable code, ahead of dynamic simulation

Extensibility/Openness. Finally, the database needs to be extensible and allow the addition of any information or metric that may be application-specific or even not currently known. An example is information or metrics from a tool yet to be developed. It also needs to be completely open and have the ability to add or remove any data with a clearly defined interface. This requirement allows any third-party tool to write data into the database or extract data, allowing the unification of data across tools from one or multiple vendors.

MENTOR GRAPHICS UNIFIED COVERAGE DATABASE

The Unified Coverage Database (UCDB) from Mentor Graphics has been architected from the ground up to meet the requirements outlined earlier. The UCDB has been the default coverage database format for storing code coverage and functional coverage metrics in both ModelSim and Questa since version 6.2. In addition, the extensibility of the UCDB has allowed test data, assertion and coverage results from Mentor Graphics Questa Formal Verification, Questa ADMS and Veloce® Emulation products to be combined. In addition to its coverage storage abilities, the UCDB also stores verification plans and test-specific data, making it a solid anchor for any verification team that intends to adopt a verification methodology that is driven from verification plans, design and/or requirements specification documents. One of the biggest verification challenges is having the ability to bring together the data and benefits from multiple verification techniques. The UCDB merge algorithms have been developed to take into consideration data from both formal (static) and dynamic

verification engines. It has the ability to combine results and report on any conflicts that may occur when comparing static and dynamic techniques, as well as allowing a static formal engine to exclude coverage from the dynamic simulation engine that is flagged as unreachable.

Leveraging the unique test-associated merging capability it is possible for a verification team to maintain a single merged database that contains merged coverage data from multiple verification runs or simulations in a regression. A record of the attributes, commands and settings of any tool are associated with each test or testcase, giving it a unique label to allow test association with coverage data. The architecture allows verification plans to be imported and linked with multiple coverage metrics or tests. This single database has enough information within it to help figure out the test(s) that incremented a specific coverage bin. There is enough information to perform test ranking (aka coverage grading) on this merged database that allows the verification team to identify the most effective tests or seeds in the case of constrained random simulation. Merge performance has the biggest impact on any ranking algorithm; to gain the most optimal results the number of merges required for what if analysis is the square of the number of tests plus the number of tests, all divided by two. This greedy algorithm makes the overall performance of a ranking algorithm very sensitive to the single merge performance due to the sheer number of merges required to gain the most optimal result. With the UCDB's unique test-association merge, this analysis can be carried out from the single merge database, bringing the necessary performance for fast and accurate test analysis. Test association provides substantial disk space savings since it is no longer necessary to keep all the individual coverage databases after a test-associated merge has been performed. Another benefit is high performance of the analysis tools. Questa users benefit from two orders of magnitude reduction in their storage needs.

Once a verification project gets underway, it becomes necessary to track project momentum by looking at the trends of metrics over time. The UCDB has been architected to support shallow or "trend" merge of multiple merged UCDB databases in order to capture trend information for coverage within the merged UCDBs. The resulting trend UCDB can be visualized via graphs, HTML

or CSV data that can be extracted from it for processing and analysis in other tools. Giving users an automated way of reducing and keeping the relevant data has the benefit of further reducing the volume of data required to analyze the project's progress.

The UCDB is extensible, scalable and open. Many different verification plan formats including Word, Frame, Excel, and Docbook can be imported into the UCDB. With a little XML and Style-sheet knowledge, a user can expand the list even further. The user can add any test-specific information in the form of an attribute value pair to the UCDB in addition to the attributes automatically captured by the tools. For the ultimate in flexibility, there is the UCDB C API. TCL-based CLI commands in tools such as ModelSim and Questa are implemented using the UCDB C API. Since the UCDB C API is open, it can be leveraged in many different interesting ways. For example, coverage data in third party tools can be extracted into the UCDB and then combined with coverage from Mentor Graphics tools and analyzed using the Verification Management tool suite in Questa. The UCDB C API could be used to read existing UCDB files in order to perform tasks such as generation of custom reports, performing custom queries or analysis.

Finally, the database requires different access methods to allow both performance and capacity. This has been solved in the fact that the database has both in-memory and streaming access to the stored data. This is extremely important because different analysis tools have different requirements. Some need the random access of an in-memory mode which allows multiple queries, while others need the performance to quickly access particular data for reporting purposes. The UCDB has been uniquely architected to achieve both modes of access and operation giving the best of both worlds.

THE UCDB HISTORY

Mentor Graphics began architecting the unified database nearly seven years ago. In early 2006 it released the first implementation, used natively by its simulation products to save coverage and assertion data. The database was developed from the start to store all the information needed to manage the verification process. The open API (Application Programming Interface) has been used to develop all the verification management and coverage tools within Questa and ModelSim, and is key to its extensibility

to other verification tools such as Questa ADMS and Questa Formal tools. This proven implementation has been used and stressed with many users' designs and environments allowing the database itself to be refined, optimized and tuned to provide the capacity and performance required by the largest of designs.

Soon after the first implementation of Mentor Graphics' UCDB was being stressed in its use on real projects, Accellera formed the UCIS (Unified Coverage Interoperability Standard) working group. This group was formed with the goal of developing a standard for the interchange of coverage between vendors. It is made up of both EDA vendors and user representatives from the largest companies in the industry. After a period of time, Mentor Graphics representatives decided to donate its technology as a starting point for the standard due to the fact that it clearly met the requirements laid down by the group. This triggered other donations from other sources. However, the UCDB API was chosen as the basis of the standard after a lengthy period of analysis of the donated technologies, giving further credence to its capabilities. With the standardization process well under way, users will start to benefit from Mentor Graphics pioneering work and database optimizations, particularly as other vendors introduce solutions based on the UCIS.

CONCLUSION

Mentor Graphics had the vision that the cornerstone of verification solutions was an optimized and unified database. The UCDB was architected and implemented to have the capabilities to allow the unification of all verification data and allow the development of very powerful verification management capabilities, which can be found within Questa today. The UCDB implementation supports a growing number of both Mentor Graphics and third party tools, and also many custom user-created analysis tools using the Open UCDB C API. An optimized unified database is definitely a reality today, particularly when using Mentor Graphics products.