

# Process Management: Are You Driving in the Dark with Faulty Headlights?

by Darron May, Manager of Verification Analysis Solutions, Mentor Graphics Corporation

## INTRODUCTION

If your car's headlights were faulty, would you even think about leaving home in the dark bound for a distant destination to which you've never been? Even if you were armed with a map, a detailed set of instructions and a good flashlight, the blackness on the other side of your windshield would still make it difficult to navigate. And even with reliable headlights and maps, you'll invariably still encounter obstacles and detours.

These hassles are nowadays mitigated somewhat by car navigation systems that tell us where we are, how far we have travelled and, so we can consider alternate routes and estimate how long it will take to get to our final destination, how bad the traffic is ahead.

Verification of SOCs and electronic systems is certainly a little more complex than road navigation. However, the process of planning what you are verifying and then constantly measuring where you are in the process is equally important whether your final destination is a swanky new hotel a few hours away or a successful tape-out by the end of the year.

A good verification methodology enables electronic closure of verification against a plan written in a human readable form, and an ability to make real measurements during verification and then link those to the original verification plan. To offer such features, the verification process requires not only the unification of different coverage metrics but also the unification of data from multiple tools and verification engines. Accordingly, data management is the foundation of any verification environment, which means that a well architected database offering both performance and capacity is critical to success.

This is the second article in a verification management series that shows how Mentor Graphics Unified Coverage Database (UCDB), donated to Accellera in 2008, has allowed powerful tools to be developed around the contents of the database. One key result of the database and its increasing use: verification engineers and managers are

now in the position of having their own navigation system for the verification process.

## GUIDING THE VERIFICATION PROCESS

All projects start with a design specification detailing what needs to be built. These specs are used to develop a strategy about what to verify to ensure that every design requirement is ultimately implemented, tested and functioning correctly. This verification plan, or testplan, is then used to guide the process of verification. Often some requirements are more important than others, though these relative rankings can change during a project either due to adjusted specs or the addition of new features. Indeed, a process guided by the verification plan needs the ability to react dynamically to changing project circumstances to ensure that acceptable quality is reached on time and with the allocated resources.

The process starts by decomposing each of the features of the specification down into test requirements. Each of the low level requirements then has one or multiple metrics associated or linked with them which indicate whether the design is doing the right or wrong thing. This, in turn, allows a coverage model to be written that is implemented within the language of choice and gives the verification team indication of where it has been. The link information from the testplan to the coverage model should be able to be put into the document itself or into the design or testbench source code.

## TESTPLAN TRACKING

A list of sections or verification requirements comprise the testplan and allow metrics to be associated directly with each tested feature. Executing electronic closure then allows this information, which can be written in nearly any document format, to be read into the database and merged with the coverage results from multiple verification runs. The UCDB has the ability to store testplans as a hierarchy and associate any combination of the stored coverage metrics or directed tests with the testplan sections. Questa

#	Title	Link	Type	Weight	Goal	Description	ENGINEER	TEAM	MANAGER	METHOD	PRIORITY	MILESTON
1	Transmitter			2	100	The transmitter is able to transmit frames in a number of different modes that need to be verified separately.	darronm	UK	Georgel	Simulation	1	30-Oct-11
1.1	Bonding_MCDE_0	cover_fsm_idle_to_neg DataTest ModeTwoTest	Directive Test Test	1	100	This mode provides initial parameter negotiation and Directory Number exchange over the master channel then reverts to data transmission without delay equalization. This mode is useful when the calling endpoint requires Directory Numbers but the delay equalization is performed by some other means (e.g. attached video codec).	darronm	UK	Georgel	Simulation	2	31-Aug-11
1.2	Bonding_MODE_1	cover_fsm_idle_to_build monitor_channel_data:CR monitor_channel_data:FAW monitor_channel_data:chana	Directive CoverPoint CoverPoint CoverGroup	1	85	This mode supports user data rates that are multiples of the bearer rate. It provides the user data rate with the full available bandwidth but does not provide an in-band monitoring function. The overhead octets are removed after the call is phase aligned. Error conditions on one or more channels that disturb overall system synchronization are not recognized automatically. Recovery from these error conditions requires manual or external intervention. This intervention is outside the scope of these procedures.	darronm	UK	Georgel	Simulation	1	31-Aug-11
1.3	Bonding_MODE_2	cover_fsm_idle_to_m2data cover_fsm_add_channel_mode2	Assertion	3	100	This mode supports multiples of 63/64 of the bearer rate. It provides an in-band monitor function as defined in Mode 3 but does not use an extra channel to replace the overhead octets required for monitoring. The user data rate is that bandwidth remaining after the insertion of overhead octets (i.e. 98.4375% or 63/64 of the total network bandwidth).	alanf	USA	Johnf	Formal	1	30-Sep-11
1.4	Bonding_MCDE_3	cover_fsm_idle_to_m3data cover_fsm_add_channel_mode3 frame_store_state/states	Directive Directive FSM	1	99	The user data rate is an integral multiple of a kbit/s including N x 56 and N x 64 kbit/sec. All channels use the same bearer channel rate. An in-band monitor function provides a continuous check for delay equalization and an end-to-end bit error rate test. (Error rate testing is accomplished by performing a cyclic redundancy check calculation on an octet sequence before transmission and testing the same sequence for errors on the receive end.) The overhead octets required for monitoring are provided by adding bandwidth (most likely an additional bearer channel) thereby	darronm	UK	Georgel	Simulation	1	30-Sep-11

Figure 1- Example of an Excel testplan.

Verification Management (VM) can read and import testplans from any file format, including Excel or Calc spreadsheets, and Word, Adobe or Write documents.

The utility can be customized to read any format of data into the database, so these test requirements can even come from an external requirements database like IBM Doors or Mentor Graphics ReqTracer. The UCDB itself unifies coverage from multiple verification engines, including analog and digital simulation, formal verification and emulation. This enables the testplan to pull together the results of all verification methods. And the testplan can also carry other planning information such as a feature's priority, the engineer responsible for verifying it and the verification method that will be used.

Attaching this kind of information to the plan allows results to be filtered in any number of ways. For example, by breaking down the verification requirements into priority categories, it's possible to assign numbers to the must-have and nice-to-have features, and those in between. This makes it possible to focus verification on the most important features first. Likewise, by adding information on engineering resources to the testplan, it is possible to track each engineer's progress in verifying features for which they are responsible. This allows managers to spot problems and shuffle resources around during the project to ensure that schedules are still met.

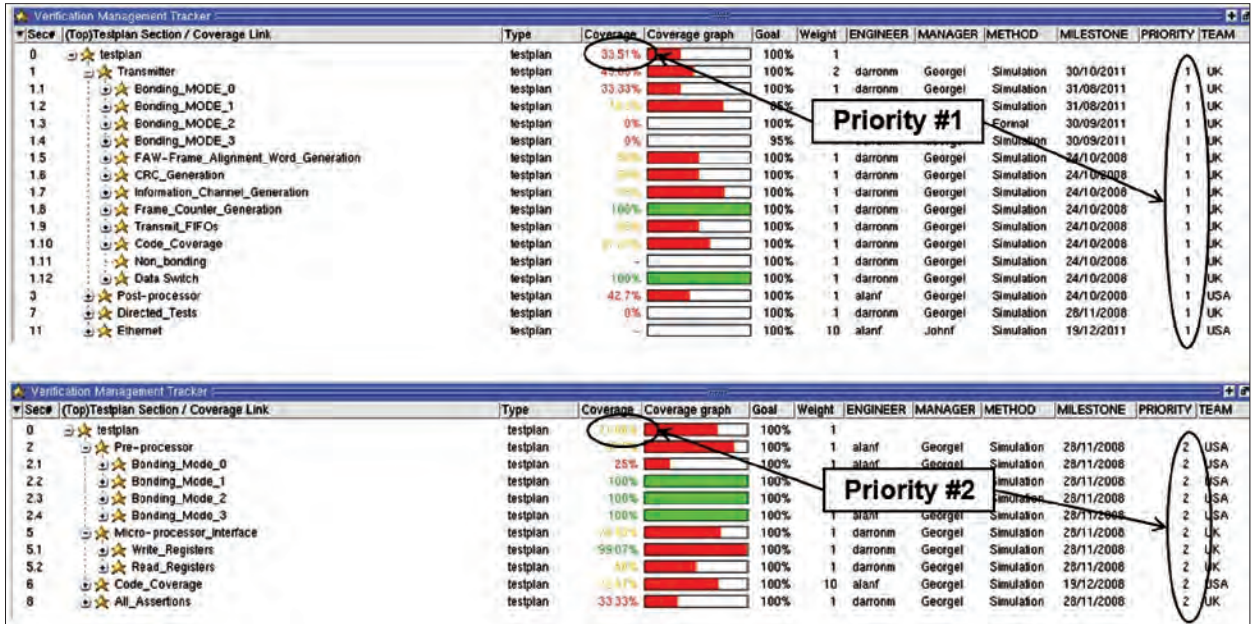


Figure 2 - Showing two queries based on priority of features.

### CONTROLLING MEASUREMENT

Because metrics from multiple sources are being combined to give an overall measurement of completeness, the testplan tracker must allow users to control calculations within the testplan document. Other attributes such as weights and goals can be attached to the testplan sections to allow further control over how the metrics are combined. By giving a higher weighting to a section of the testplan, it's possible to increase its contribution to the overall testplan coverage calculation. If a more automated way of weighting sections against each other is required, then other coverage calculation algorithms can be used. (An example is a calculation based upon the number of linked objects.) It's also important to bear in mind that not all coverage is implemented when a testplan is first generated. Thus, the fact that a coverage object doesn't yet exist needs to be taken into account. By marking links within the testplan as yet to be implemented, Questa testplan tracking automatically generates coverage objects as part of the merge process. Until they are implemented, these objects then show as being uncovered in the overall testplan coverage data.

### REUSE

It's extremely common today for the latest project to use IP or VIP from a past project or external source. Verification of this IP also requires a testplan, thus the need to reuse testplans within new projects. For example, all Mentor Graphics Questa VIP is shipped with testplans that can be read into the Questa testplan tracking tools. This means that support for hierarchical testplans is required not only to allow reuse but also to make it possible to break down large testplans into more manageable pieces. Because different users and IP vendors rarely provide their testplans in the same file formats, it also must be possible to mix and match the testplans from any source document. Questa testplan tracking provides the ability to add links within a document to refer to any other document type. This means that an Excel spreadsheet can have a testplan section that refers to a Word document or vice versa.

Another important requirement that enables reuse is the ability to parameterize settings that vary based on environment. For example, some links embedded within a testplan may have path settings for a particular coverage object. If a particular setting refers to a path in a piece of IP, then this path will change when used in different

environments. Likewise, if a specific use of a given block of IP means that a coverage object will never be used, then the object needs to be excluded. Accommodating these kinds of changes require parameters that can be set based on testplan usage.

## IMPROVING TESTPLAN ENTRY

Forcing a user to capture their testplan with a particular entry method or proprietary editor creates heartburn. The good news is that building a proprietary testplan editor is not the only way to make capturing testplans easier. Most document applications like Excel, Calc and Word have their own extension languages that make it possible to integrate custom features. Questa testplan tracking provides add-ins that enable creating testplans with the application of choice. For example, the Excel add-in has a testplan creation wizard to make generating a testplan template very straightforward. Working in Excel allows for easy addition and deletion of sections and user attributes, and also automatic re-numbering and formatting. And the Excel testplan in this example interfaces directly to the UCDB for information on the objects available for linking.

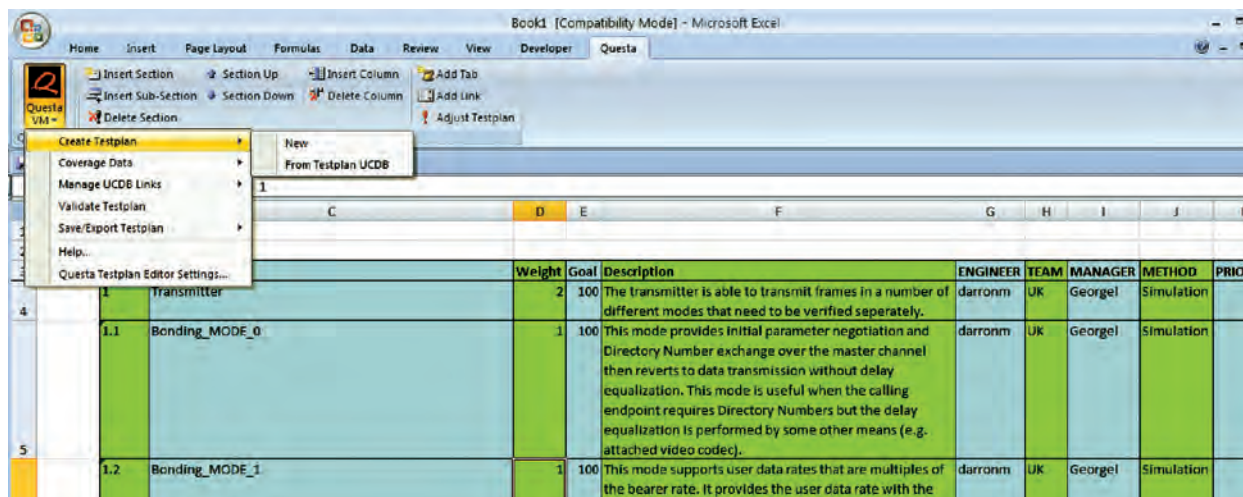
Testplan validation can be run to ensure the plan is in the right format before importing it to Questa and it is possible to save the testplan directly to a UCDB file, thus completely

automating the import process. Once coverage data is available, the add-in is able to annotate coverage information back into the spreadsheet. To ease adoption it will also regenerate a spreadsheet directly from a testplan that's already been imported.

## COVERAGE MANAGEMENT

Another important part of the process is how coverage is captured over regressions throughout the project. The UCDB is used to capture both coverage and test information from single verification runs across all engines. Then UCDB utilities combine, merge or optimize the data via ranking. A merged database can be used to answer all kinds of queries about the effectiveness of a particular testplan section, while at the same time reducing the space needed to store all the single verification runs. The database still holds information such as how and where the test was run, and how long it took. This allows for optimization of test suites by using the ranking utility to either reduce the number of simulations (by finding redundancy) or find a particular set of tests that will run in the shortest possible time while still achieving a particular coverage goal. Data management over multiple regressions and the overall project duration are flexible due to the fact that the data is available on a test by test basis.

Figure 3 - Excel add-in making creation easy.



Action	Tests	Methods	Total Coverage	Tool Runtime
Sort	35	31 Simulations	NA	NA
		2 Formal		
		2 Emulation		
Merge	20	16 Simulations	67.08	17:38:24
		2 Formal		
		2 Emulation		
Rank Fewest	8	6 Simulations	67.08	6:47:46
		2 Formal		
		0 Emulation		
Rank Quickest	9	5 Simulations	67.08	3:47:18
		2 Formal		
		2 Emulation		

Figure 4 - Finding the best combination of tests.

Normally data is combined from verification runs of the same source code, making it straightforward to locate the same coverage objects and calculate total coverage. When changes are made to the source code and different coverage models are combined, the merge process must decide what to do when objects do not match.

Questa provides either a union merge or a base merge of data. A union merge combines all coverage from all source databases and can lead to a disconnect between the coverage data and the current source code. Great care should be taken when combining data this way. One reason: code coverage, with its dependencies on code lines, becomes stale very quickly as source code changes. Merging data from the same source code builds is recommended, though using a second merge method can also help with the problem of stale coverage. Base merging allows a base database to hold the objects that will be merged from all other provided databases. This allows the

current database represented by the source code to be used as a filter at merge time. If an object in the base database matches the database being merged, the object will be included. This method is beneficial for functional coverage merging because it excludes old coverage and also helps to merge slightly different code coverage models.

### TREND ANALYSIS

As the project progresses problems can arise with data storage requirements. Keeping the results from multiple regressions over time requires vast amounts of storage. Consider, too, the changes in the source code over time, which can make it very hard to combine the detailed information. What's required is a different kind of merging that is suited to looking at the progress of coverage over longer periods of time. By taking snapshots of relevant data across regressions, it's easier to spot higher level trends even without keeping all

the details of every single test. The UCDB database has been formatted to allow the user to decide how to manage the data by combining runs from regressions. Then, with a trend merge of the database, it's possible to take snapshots from these regressions over time. The trend merge is a shallow merge of the data that extracts coverage on the basis of design unit, testplan and functional coverage, or instance by instance. This reduces the amount of data that needs to be stored for each regression run. It also makes the stored numbers less susceptible to changes in source code because only metrics for higher level objects are stored within the database.

Of course other metrics within the verification process can be useful for tracking other tasks and milestones. Being able to track metrics within other parts of our verification methodology and view those with the coverage data can give excellent evidence of progress being made towards tape-out. For example, seeing the number of open, closed and assigned bugs over time gives an indication of whether bugs are still being found. Looking at the number of lines

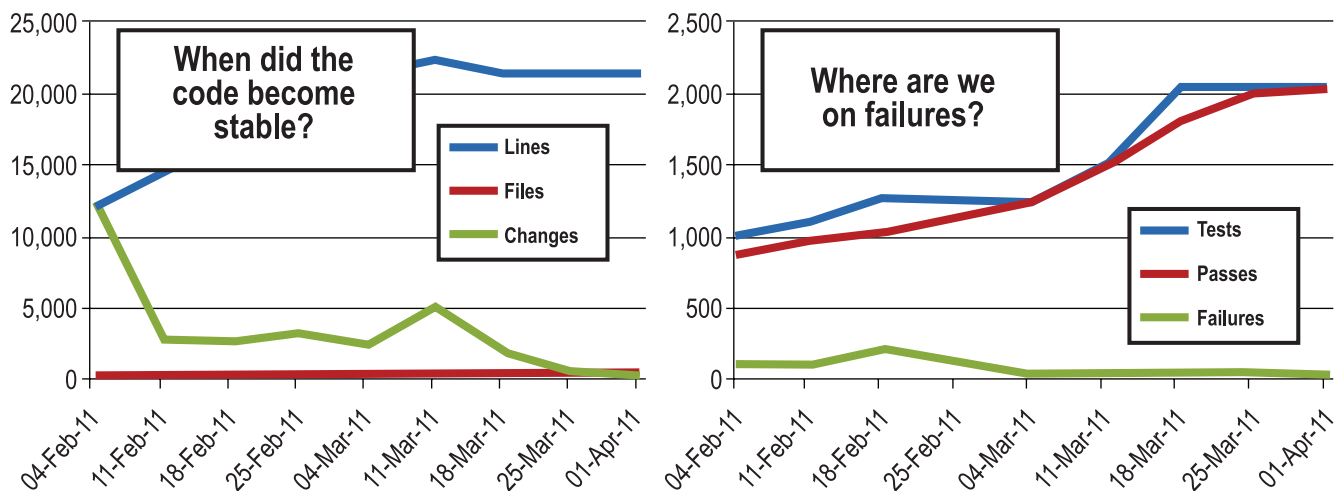
of source code and the number of lines that have changed from week to week can suggest how design stability is changing. Monitoring the number of tests that are run, the type of tests, and the number that have passed and failed over time also gives us a measurement of completeness. The trend merge allows any user metrics to be added to the UCDB and trended over time with the coverage metrics.

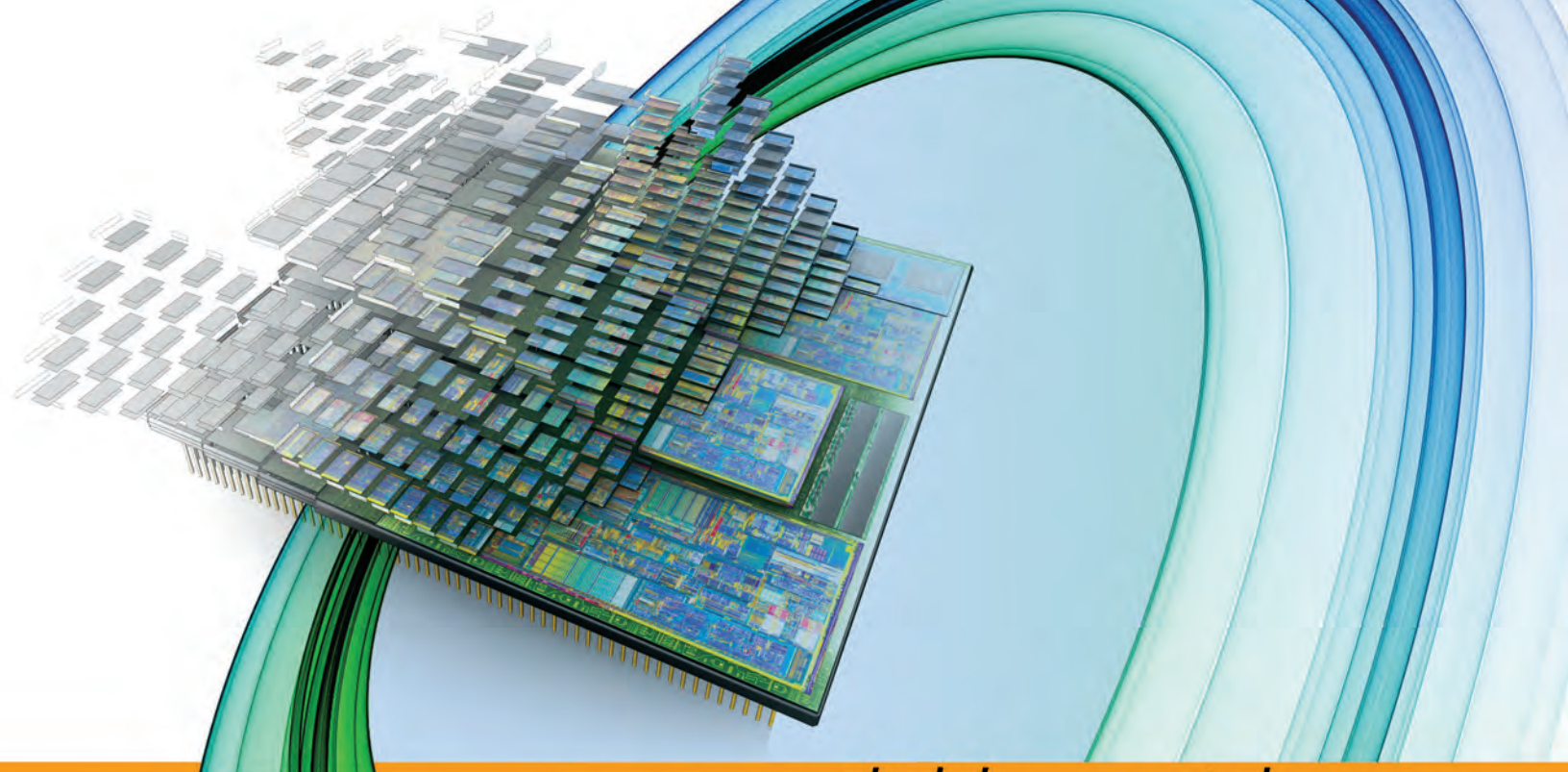
## CONCLUSION

Mentor Graphics anticipated that the cornerstone of verification solutions would be an optimized and unified database. Thus, UCDB was architected and implemented to allow the unification of all verification data and the development of powerful verification management capabilities, all of which can be found within Questa today. These capabilities include testplan tracking to allow electronic closure of the process with a plan, utilities that allow data to be combined and optimized when generated from multiple tests and engines, and the ability to reduce data over time while still exposing data needed to see project progress.

Today, you wouldn't leave home without your car navigation system to guide you to your destination and give you real-time location information. Questa VM gives you the same visibility and guidance for your verification process.

**Figure 5 - Trending metrics over the duration of the project.**





# *verification* HORIZONS

Editor: Tom Fitzpatrick  
Program Manager: Rebecca Granquist

Wilsonville Worldwide Headquarters  
8005 SW Boeckman Rd.  
Wilsonville, OR 97070-7777  
Phone: 503-685-7000

To subscribe visit:  
[www.mentor.com/horizons](http://www.mentor.com/horizons)

To view our blog visit:  
[VERIFICATIONHORIZONSBLOG.COM](http://VERIFICATIONHORIZONSBLOG.COM)

**Mentor**  
**Graphics**<sup>®</sup>

[www.mentor.com](http://www.mentor.com)