VHDL Model Verification and System Life Cycle Support

Steven L. Drager, James P. Hanna, Robert G. Hillman
Electronics Reliability Division
Rome Laboratory
Rome, NY 13441-4505

ABSTRACT:

Industry standards, such as the VHSCIC Hardware Description Language (VHDL)[1] and the Waveform And Vector Exchange Specification (WAVES)[2], are critical elements in being able to integrate the design and test cycles. When performing top-down designs based on these standards, the descriptions necessary for system life cycle support are obtained and captured during design. Together, VHDL and WAVES provide a powerful mechanism for concurrent engineering practices by allowing digital stimulus and response information to be freely exchanged between multiple simulation and test platforms. WAVES is defined as a syntactic subset of VHDL, and as such, can be simulated against the VHDL model during design to verify functionality and timing. Further, after the hardware is built, the same WAVES data may be ported to the test environment for electrical test. This process assures that the same source stimulus and timing that were used during design are applied during electrical testing. VHDL and WAVES design validation methods are a major advancement in assuring that digital electronic systems are truly designed for the goal of two-level maintenance and on-equipment fault diagnosis. The authors will discuss the IEEE WAVES re-ballot requirements in terms of a complete top-down design methodology. This paper will also discuss the current development of WAVES based software tools and products to support VHDL model verification.

INTRODUCTION:

As microelectronic components and printed circuit boards become more complex, the costs associated with supporting these products continue to increase. It is estimated that testing and support comprise more than 70% of the life-cycle cost of today's complex microelectronics. With complexity ever increasing, it is more critical than ever to reduce the cost of testing and supporting electronic systems.

System design practices which bring test engineers and design engineers together during the design process are becoming more commonplace. This integration assures that adequate attention is paid to Design for Testability (DFT). This Integrated Product Design (IPD) approach proves to be effective not only in assuring due consideration to testability issues, but also in improving diagnostics and reducing back-end testing support cost.

One of the keys to improving the IPD process lies with capturing and re-using the original design data throughout the product life-cycle. Industry standard representations of design, test, and diagnostic information are essential to provide maximum data portability and re-use. This data, along with supporting tools, will provide the necessary automation links between the design and test engineer.

WAVES FOR DESIGN AND TEST:

In order to support the life cycle of electronic systems, the DoD, as well as the commercial community, need a common format for capturing the stimulus and response information for digital microelectronics. This information is required not only for efficient life-cycle support, but test verification which must be performed during design to verify that functional requirements are met. This same test verification data is also required during electrical testing to insure the correct functional operation of the
electronics. WAVES, the Waveform And Vector Exchange Specification, is the IEEE language developed for capturing and exchanging input stimulus and expected response information (test vectors/signal histories) for digital microelectronics. The WAVES standard is intended to be the vehicle to facilitate this bridge of test stimulus between the design and test processes. The goal of the WAVES standard was to permit simulation histories and test vectors to move freely back and forth among simulation and test environments in a common, non-proprietary format.

The WAVES standard, however, has been slow to catch on within the VHDL community for several reasons:

1. The original WAVES standard package implementations developed were intended to be a prototype of the specified functionality. VHDL simulation efficiency was therefore not addressed in this implementation.
2. Due to the perceived complexity of WAVES, there have been inadequate attempts to publicize the benefits of using WAVES for both VHDL top-down design and electronic test.
3. Little in the way of user documentation exists to help potential users understand how to write WAVES data sets and integrate them with VHDL test benches for design verification.
4. Finally, the last reason for the limited use of WAVES is the lack of EDA tool support for WAVES.

All of these limitations are currently being addressed and the solutions are beginning to emerge, as will be discussed throughout the remaining sections of this paper.

**WAVES RE-BALLOT / VHDL FOCUS**

In order to be able to focus on VHDL integration, the WAVES working group has formally joined the Design Automation Standardization Committee (DASC). A new Par (P1029.1) entitled, "Standard for VHDL WAVEFORM and Vector Exchange (WAVES) to Support Design and Test Verification," was submitted and approved September 21, 1995.

The scope of the working group, as defined by the PAR, is two fold. First, to bring the 1991 WAVES Standard into alignment with the VHDL 1076-1993 Standard. WAVES, as an application of VHDL, must exist in alignment with the VHDL-1993 Standard. Second, to improve the interoperability of the WAVES-VHDL interface. Users of WAVES have identified deficiencies in the usability of WAVES in VHDL simulation and it is the intent of this working group to address and correct these issues.

The group is currently re-evaluating the original language requirements and addressing the WAVES standard relative to efficient VHDL simulation. All of the VHDL93 changes have been looked at, the impact to WAVES studied, and appropriate actions proposed. The areas impeding WAVES usability have also been researched with changes proposed. Finally, development of efficient WAVES Standard Packages has been studied and changes proposed. Efficient simulation is a driving principle behind the implementation of all changes proposed by the working group.

As mentioned above, the original standard package implementation was only intended to be a prototype of the specified functionality. These prototype packages are, however, the only implementation that currently exist. These packages must be modified to handle the most optimal implementation possible for VHDL simulation. A new WAVES standard package proposal is planned for completion in early 1996 that addresses both VHDL simulation execution and data space efficiency. The current implementation creates data structures which are never used in simulation, but exist only to support vector translation to ATE (automatic Test Equipment). This low utilization is being evaluated relative to the additional burden of implementation.

There are many enhancements being integrated into the current proposal. The following paragraphs discuss some of the important issues related to the VHDL-WAVES interface.

WAVES logic values define the names of legal events (logic levels) that may occur on a given signal within the WAVES data set.
The syntax of the current implementation is restricted to an enumerated type where neither character literals nor subtype declarations are allowed. The main intent of this restriction was to avoid confusion between the logic values and the external file pin code strings used to apply waveforms. However, these restrictions have had the opposite effect of making the integration with a VHDL model too clumsy. The proposed change will allow the use of character literals as well as subtype declarations. For example, a subtype declaration for Std_1164_ulogic would be an acceptable logic value definition. This proposal has minimum impact on the WAVES standard, but will greatly enhance the WAVES VHDL interface. This proposed change will support direct WAVES logic value assignments to a VHDL model's compatible signals and logic types, such as Std_1164_logic.

The interface between VHDL and WAVES is established via a signal defined in the WAVES LRM as the WAVES PORT LIST (WPL). The proposal for use of subtypes for logic values greatly simplifies the implementation of the WPL. The WPL signal data type was never intended to be defined under the current WAVES LRM. The intention was for the WPL to be implementation specific, therefore, the existing packages inefficiently define the default interface. The current data structure limits the implementation by utilizing a single fixed record. This results in several modeling inefficiencies. The data structure itself is larger than necessary and the WPL record cannot be sub-sliced using array indices for signal associations. The proposed change utilizes a logic value array, so that signals can be directly assignable sub-slices of the WPL signal.

The proposed WPL also enhances VHDL usage by supporting a concept the working group refers to as waveform tiling. Multiple waveform generators (procedures) can be easily combined to provide a composite set of waveforms. For example, a waveform generator procedure defines the multi-clock inputs to a CPU and a second waveform generator procedure defines the data to be applied.

The adoption of a subtype definition for logic values has an additional benefit. Any user defined logic value subtype may be applied to the model during simulation. During performance modeling, the common practice is to use a structured record, referred to as a token, for the values to be assigned to the signals. This performance modeling practice may be directly supported by the proposed logic value change. This added WAVES capability in essence falls out for free.

**VHDL MODELING HIERARCHY:**

VHDL ([IEEE Std 1076]) is a formal notation intended for use in all phases of the creation of electronic systems. It supports the development, verification, synthesis, and testing of hardware designs, the communication of hardware design data, and the maintenance, modification, and procurement of hardware. The following definitions define the four levels of VHDL models in our top-down design methodology.

*Performance Model:* A simulation model that uses abstract data types and enumerated types to convey the functionality of the design, without specifying implementation details. These models are useful for evolving a specification before doing any detailed design.

*Behavioral Model:* A simulation model that uses a bit value interface for the entity declaration of the design, without requiring any specific implementation details. There will be multiple iterations of behavioral models, where each iteration will incorporate added design detail, before the refinement to the RTL model will be made.

*RTL Model:* A simulation model which is concerned with the detailed verification of data and control sequencing both within and between blocks. These models are described by functional operators and procedural constructs.

*Structural Model:* A netlist representation that specifies which
hardware components are to be used and how they are to be interconnected.

Our top-down design methodology begins with a VHDL model which uses Boolean, numeric literals, and user specified enumerated types. It contains a single entity/architecture pair and usually uses only one process. This high level model will be used to verify the functionality of the design. This model will eventually be expanded to a behavioral model which includes the individual blocks of functionality. Each block may be described by a separate behavioral model. This behavioral model will define the interface requirements between the functional blocks, as well as perform the first partitioning of the design. Further partitioning within each block will occur as we proceed to the RTL model. The continued partitioning will help in managing the complexity of each block, reducing the design into manageable portions and insure a successful design.

**VHDL/WAVES Design Methodology:**

The IEEE WAVES re-balloting activity is targeted at supporting a complete top-down design methodology. The methodology for applying WAVES for design and test to the VHDL modeling process at each of the levels of the design hierarchy is represented graphically in Figure 1. The design process starts with a high level abstract performance model. A WAVES token based behavioral testbench is used to refine the behavior of the VHDL performance model. The token interface is a record which contains abstract data type elements passing a parameter to the model. This behavioral testbench may read an external pneumatic type instruction file(s) or it may use procedural algorithms directly coded in.

At the next level of refinement for the behavioral model, the token interface is replaced with a bit type logic value. Translation between the algorithms or instruction file commands and the bit interface for the model is easily performed. The WAVES logic value interface in this case is simply a subtype definition to a bit type, i.e. STD_1164_logic. It is important to note that the WAVES bit level dataset applies the same instruction file(s) or algorithm and translates the instructions to scalar signals.

![Figure 1. VHDL - WAVES Design and Test Flow.](image)

This same test bench may then be used to verify the behavioral, RTL-level, and gate level VHDL models. When we reach the end of the design verification process, a WAVES testbench can be used to generate and translate the dataset into a new test set based on the WAVES external file format. This converted dataset should be reapplied to at least the behavioral model for regression analysis. In addition, any fault coverage tests should be added. An important point to remember is that these tests need to be referenced separately, since they only apply to the gate level models.

The final WAVES dataset is then translated into a specific tester language. Once fabricated, the electronic parts are tested against the translated data set, in essence the same test used during design. The methodology for using WAVES throughout the design and test process assures that the same verification data is applied across the various levels of design and test in a consistent manner. This methodology also allows rapid and consistent regression testing to be applied.
throughout the hardware development, fabrication, and test process.

**LIFE-CYCLE DATA USAGE**

The use of VHDL in a top-down design process provides the benefits of modeling at multiple levels of abstraction, technology independence, life-cycle documentation and validation through simulation. The benefits of using WAVES for design verification within a top-down design methodology are that it can greatly reduce the test generation cost and aid in controlling the cumulative cost of the design and test cycles encountered during a product's life-cycle.

VHDL and WAVES allow for the elimination of duplication of effort throughout the life cycle of a system. When a system is first designed, it is simulated against a set of tests to prove that the design implements the specification. This same stimulus provides the starting basis for testing once the design has moved into manufacturing and then into the field. The stimulus used in manufacturing and in the field are augmented with additional tests, but savings occur by not having to derive the base test set from scratch. By archiving the complete VHDL and WAVES model library, tremendous cost savings may be realized during reprocurement and system upgrade.

If one faces a reprocurement of a system, the VHDL and WAVES methodology greatly reduces the cost, as the design does not have to begin from scratch. By archiving all VHDL and WAVES model files during the initial system design, the reprocurement manufacturer need only fabricate the design. The design and test stimulus have already been completed, savings thousands of hours of development time.

When the time comes for system upgrade (i.e. a new architecture, new technology or miniaturization), reuse of the existing VHDL and WAVES archives play a tremendous role in driving down the upgrade cost. In some instances the redesign may be able to begin with the existing performance model and test sets, while other products may utilize various behavioral models and test sets. By archiving the VHDL and WAVES models of the original design, both the cost and development time are greatly reduced.

**VHDL BASED TOOLS FOR TEST**

In addition to the re-balloting activity, other interesting activities surrounding the WAVES standard are beginning to build. Universities and commercial Electronic Design Automation (EDA) companies are beginning to develop tools and documentation for easing the use of the WAVES standard.

Many of the initial design application problems encountered by the VHDL community (e.g. efficient simulation, synthesis, etc.) have been solved and commercial tools have been made available to the VHDL design community. Several "companion" standards to VHDL have been (and are being) developed. WAVES is one such standard.

The next major focus of the VHDL community will be in the area of test and design for test; WAVES will play an important role in this area. It is extremely important that the VHDL community become more aware of the use of, and benefits of using WAVES in the top-down design cycle. It is essential that we continue to educate this community about WAVES. A number of software tools that use the WAVES standard are under development. Some of these tools are being developed under the sponsorship of the Air Force by Rome Lab and Wright Lab. Others have been developed under research funding provided by universities.

The University of Cincinnati (U.C.) has developed a set of WAVES test bench compilers that integrate the development of WAVES into the VHDL design process. These tools were developed to automate the design, development, and testing of Multi-Chip Modules (MCMs) from high-level behavioral VHDL models. This tool set includes a behavioral-level test generator, a WAVES test bench compiler, a multi-chip test bench compiler, and a boundary scan test bench compiler. Together these tools provide a seamless VHDL and WAVES design environment that will provide a
much needed automation capability and will greatly increase the use and acceptance of the WAVES standard.

Rome Laboratory, under the Small Business Innovative Research (SBIR) program is sponsoring the further development of the U.C. WAVES tools and the development of a hyper-media, interactive WAVES design assistant tool. This SBIR program was awarded to MTL Systems, Inc., of Dayton OH. MTL has teamed with UC to develop and commercialize these tools and the hyper-media design assistant. In addition, MTL is developing a hard copy version of the hyper-media design assistant that will be made available as a WAVES User's Guide.

Further development of the U.C. tool set will establish the tools as a standalone tool set that can be used with any VHDL design environment to integrate the use of WAVES into the VHDL design cycle from high-level behavioral modeling through system partitioning and synthesis.

The on-line hyper-media design assistant will provide the much needed documentation on the development and use of WAVES data during the design process. This tool will guide the user through the development and application of WAVES for design verification in any VHDL design environment.

In September of 1995, Rome Laboratory awarded a contract to develop a prototype tool set that would provide the capability for fault simulation and Automatic Test Pattern Generation (ATPG) for VHDL models. This tool set is being developed at the Center for Semiconstom Integrated Systems at the University of Virginia.

The tools developed under this effort are based on the Advanced Design Environment Prototype Tool (ADEPT). This tool was developed under the joint sponsorship of NSF, ARPA, NRL, Honeywell, NASA Langly, Union Switch and Signal, and the ARMY. ADEPT is a collection of tools that together form an environment that supports the unified modeling of systems. The ADEPT tools allow users to model systems across the entire design hierarchy space, from gate-level to performance-level. The tools automatically generate and simulate complete VHDL models. This tool set will accept vectors described in the WAVES format and use them in the performance of fault simulation of VHDL models. The output of the ATPG tools will be WAVES test patterns suitable for fault simulation and translation to a variety of Automatic Test Equipment (ATE).

Wright Laboratory, under a Phase I SBIR is sponsoring the development of a WAVES compiler/generator interface to the Ikos systems VHDL simulation tools. This SBIR, awarded to Ikos Systems in mid 1995, will provide a high performance standard stimulus/response format interface to a commercially available VHDL simulation environment. This marks the first commercial integration of the WAVES standard into an existing VHDL simulation environment.

The WAVES compiler/generator will form the interface between the Ikos high performance compiled binary stimulus language used by the Ikos mixed-level simulator. The WAVES compiler will accept WAVES compliant data sets and compile them into a binary format that can then be linked to the simulation kernel for high performance simulation. The WAVES generator will generate WAVES compliant data sets from the existing binary stimulus/response data currently used by the Ikos VHDL Simulator. This effort will yield a VHDL-WAVES tool environment that has the capacity and performance required for large, complex electronic system designs.

Rome Laboratory has developed a prototype tool set to automatically generate complete WAVES-VHDL testbenches for complex VHDL circuit designs. This tool provides a complete test set for verifying complex digital circuit designs and automatically generates IEEE compliant WAVES and VHDL source code. The test sets generated by this tool can be applied during VHDL design verification and after fabrication during electrical test. This tool significantly improves digital electronics life-cycle support capabilities by automating and more tightly integrating design and test activities.
The X- Windows Graphical User Interface (GUI), developed using TCL/TK, simplifies and enhances the usability of the tool set. This GUI also provides a single interface for user interaction with the tool set. The common GUI interface to the tool set is referred to as xstb. The xstb tool set has several features that aid the user when using WAVES and VHDL together. The tool generates WAVES packages that are utilized in defining the model specific WAVES data set. The tool also can perform a syntax check on the external pattern file that is to be utilized with the WAVES data set. This check verifies conformance to the WAVES external file syntax.

Finally, the tool generates a self-monitoring WAVES-VHDL test bench which wires the WAVES waveform data set and the model together for design verification. The main input to the tool set is the models' entity declaration. The entity declaration is parsed by the tools and used to define the appropriate signals and data structures for creation of the WAVES data set and the VHDL test bench code.

REFERENCES


CONCLUSIONS

Together, VHDL and WAVES can be the main components in eliminating duplicative re-generation cost and effort. The archived design and test information required throughout the various phases of a products' life cycle can be reused many times.

The WAVES re-ballot activity is actively pursuing a revised standard that effectively provides a standard test interface for VHDL. As the re-ballot progresses, interest and activity surrounding the WAVES standard are beginning to build. Universities and commercial Electronic Design Automation (EDA) companies are beginning to develop tools and documentation for the use of the WAVES standard.

Industry acceptance and the commercialization of useful tools for WAVES is becoming more probable with the current developments that are already underway. The VHDL focus of the re-ballot and tools currently under development will provide a great deal of support for integrating VHDL and WAVES into a seamless design through test environment.