

Comprehensive Test Suite for Validating Mixed Language Support

Key Advantages

- Backed by Interra's field-proven expertise in developing HDL-based test suites for VHDL and Verilog
- Developed in partnership with significant EDA majors
- Conforming to accepted definition and interpretation of the languages
- Providing an unbiased quality analysis of EDA tools
- Comprehensive validation of mixed language styles

Highlights

- Over 700 test cases along with test benches
- Golden output for comparison
- Detailed test plans with cross-reference to test cases
- Well organized test cases highlighting testing objectives
- Both positive and negative test cases

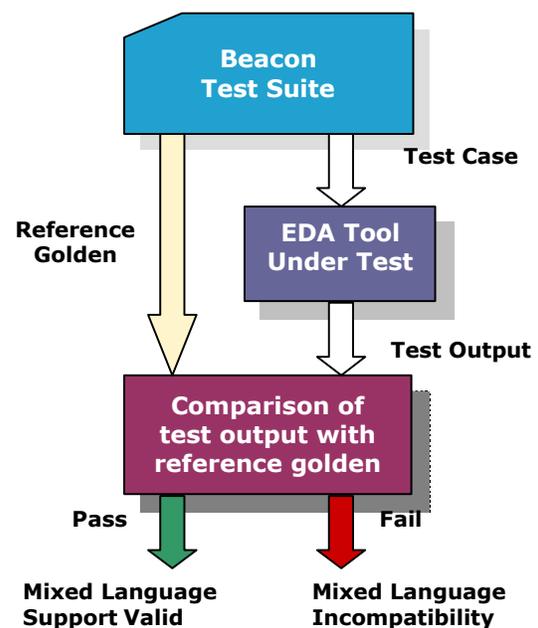
Addressing the needs of EDA tool developers to quickly evaluate the quality of their products, Interra's Beacon-MX delivers a comprehensive test suite to validate mixed Verilog/VHDL support in EDA tools.

You can use Beacon-MX to validate EDA tools developed using mixed designs: mixed instantiations, data transfer between mixed interfaces, port mapping, and parameter mapping. In addition, you can characterize EDA tools for mixed language support across various VHDL to Verilog or Verilog to VHDL interfaces.

Enabling you to discover mixed language incompatibility early in the product development and testing life cycle, Beacon-MX offers:

- Reduced development costs and time-to-market EDA products
- Development of standard-based products
- Precise evaluation of bugs and errors in the product
- Measure of product quality
- Unbiased feedback on product quality
- Regression tests for quality assurance

The test cases and test benches can be applied to the EDA tool under evaluation and results can be compared with golden reference that is provided with Beacon-MX.



The Beacon-MX Features

Comprehensive Test Suite

Beacon-MX test suite validates mixed language interface.

These test cases cover different ways in which Verilog or VHDL design units can be instantiated in other languages. The test cases also cover transfer of data from one language to the other. The test cases check for mapping of data types from one language to another. The test cases also check various VHDL port map style mapping to that of Verilog.

Test cases also include various combinations of VHDL generic and Verilog parameter to pass values. A set of test cases uses defparams to override VHDL generic values.

Sandwich cases with Verilog-VHDL-Verilog and VHDL-Verilog-VHDL check for correctness of interface data transfer across multiple language boundaries.

Negative test cases check for behavior of the test tool in case of erroneous interfaces.

Test cases are distributed as follows.

Language Construct	VHDL Top	Verilog Top
Port Mapping	79	134
Parameter/ Generic Mapping	16	113
Sub unit as library/ special instance	11	11
Component Instantiation	12	
Configuration Specification	26	
Direct Entity Instantiation	43	
Incremental Binding	30	
Incremental Binding (N-level language nesting)	29	
User Defined Data Type Mapping	15	
For/ If Generate Constructs	8	13
Cross HDL Id Mapping	38	10
Instance Array		4
Sandwich Cases	12	13
Miscellaneous Cases	1	2
Complex Port Expression	15	
Generic Value	38	
Port Expression	44	
UDP Instantiation	1	
Total	418	300

Well Documented Test Plans

The test plans describe all test objectives and are categorized by sections.

Top-design Based Organization

Test cases are organized based on VHDL/Verilog top design units instantiating a mixed design unit.

Category-A: VHDL Top

The VHDL Top category includes test cases with VHDL as top entity instantiating Verilog modules. Test cases check for various port map styles, data types, generic values, and configurations. Sandwich cases are also present which include Verilog instances instantiating another level of VHDL.

Category-B: Verilog Top

The Verilog Top category includes test cases with Verilog as top module instantiating VHDL entities. Test cases check for various port sizes, parameter values, and defparams providing VHDL generics values. Sandwich cases include VHDL instances instantiating another level of Verilog. Test cases also check parameter setting across multiple language boundaries.

Test Benches and Reference Golden

Provides test benches to instantiate test cases and apply vectors on inputs. Outputs are captured after an appropriate interval and written on to a file. Reference golden output is also provided for all the test cases. You can easily apply the test bench to the test tool and compare the outputs.

Sample Test Case

```
--** Purpose:      Named association with all
                   the scalar ports connected.
--** TestPlan:     Sections 1.1.1.2.1
--** Status:       SIMULATION_SHOULD_PASS
--** Assumptions:  GOLD is generated using
                   vsim(5.6).

*****
`timescale 1ns/1ns

module NamedAssociation1_top (In1, In2,
Out1) ;
    input  In1, In2 ;
    output Out1;
    wire  temp;
    NamedAssociation1 VhdlInst ( .In1(In1),
    .In2(In2), .Out1(Out1)) ;
endmodule
```