EFFICIENT APPROACHES TO TESTING VHDL DSP MODELS[1]

James R. Armstrong, EE Department, Virginia Tech
jra@vt.edu

ABSTRACT

Generation of test benches for large DSP behavioral models is a complicated, labor intensive task. Also, tests generated manually satisfy no formal definition of completeness. To address these needs, high level approaches to test bench development are employed which relieve the modeler of the details of this task. CASE tools are used to develop the test bench VHDL code, i.e., state machine behavior is specified with Ilogix Express VHDL and sensor behavior with Comdisco SPW. An intelligent interface prompts the user for high level test bench information, and inserts this information into the test bench code. The intelligent interface also allows the user to specify and control file I/O as a data source. Conceptually speaking, two approaches are being explored: 1)behavioral - the CASE tools develop complete high level models of the test bench, and 2)structural - a library of primitive components is developed so that a conventional schematic capture tool, e.g., Synopsys Graphical Environment, can be used to construct the test bench.

1. INTRODUCTION

A major focus of the RASSP program, and other efforts concerned with the rapid development of custom signal processors, is the development of VHDL models for signal processing algorithms and hardware. Like all simulation models, they must be thoroughly exercised to prove that they exhibit the correct behavior. However, development of model tests and the corresponding VHDL test bench is a labor intensive task. Moreover, model tests generated manually satisfy no formal definition of completeness. Thus there is a critical need to develop high level approaches to development of test benches that relieve the modeler of the details of this time consuming process.

In this paper we describe an approach to test bench development which has the following features:

1) Test bench code elements are developed using high level CASE tools. Specific to the RASSP environment, we use Ilogix Express VHDL to specify state machine behavior and Cadence/Comdisco SPW to specify signal processing behavior.

2) During model simulation the test bench is under control of an intelligent interface which provides the test bench with values for test bench generics and specifies the high level characteristics of model inputs or controls the reading of files of data reflecting the nature of those same input characteristics.

3) Test benches can be developed either behaviorally or structurally. With behavioral development, the CASE tools develop complete high level models of the test bench. With structural development, a library of primitive test bench components and a schematic capture tool is used to construct the test bench.

2. BASIC TEST BENCH REQUIREMENTS

Figure 1 shows a basic test bench. The Stimulus Generator drives the Model Under Test with test vectors. The model response is compared with the expected response by the Comparator and appropriate Go/No Go signals are generated. Two types of feedback are possible. First, the Model Under Test can feed it's state

[1] Material in this paper was previously presented at the 1st Annual RASSP Conference, Arlington, Virginia, August 16, 1994.

0-7803-2431-5/95 $4.00 © 1995 IEEE

back to the Stimulus Generator, allowing interaction between these two elements. Second, Comparator signals can be fed back to allow adaptive testing, i.e., changing of test modes when a certain response has been observed.

When the Model Under Test is a VHDL model, the test bench is also coded in VHDL. This code applies inputs to the MUT using signal assignment statements or file I/O. In addition, state machine behavior must be coded if the Stimulus Generator interacts with the MUT.

3. TEST VECTOR GENERATION

An important consideration in test bench development is: how are the test vectors that drive the model generated? Again, these could be developed manually, but for large models this is an arduous task. Recently a number of approaches to automatic test generation for behavioral models have been developed. In one approach, *model based* test generation, the model itself is used to generate the tests. There are two types of model based test generation: model perturbation with high level faults [1] and I/O path sensitization[2]. A second approach to test vector generation is *environment based* as it uses the environment surrounding the model under test to develop the test vectors. In the RASSP work we employ environment based test generation because the type of inputs experienced by the signal processing models are a complicated function of the model environment [3] and could not be directly generated from the model itself.

As an example of a complicated environment, consider Figure 2 which shows a stimulus generator for an Infrared Search and Track (IRST) signal processing model. The environment consists of four sub models: platform, target, environment, and sensor which function as follows:

1) Platform - models the motion of the platform that the signal processing computers are mounted on.

2) Targets -models target motion and interaction between the target and the platform.

3) Environment[2] - models the effects of clouds, clutter, and noise.

4) Sensor model - Combines platform motion and target and environment information to produce the sensor data which are the test vectors for the model.

It is the object of our work to develop methods of environment based test generation for situations such as this.

4. SYSTEM DESCRIPTION

Figure 3 shows a system for test bench development. A behavioral model of the test bench can be developed using a CASE tool, Ilogix Express VHDL[4]. Similarly, a structural model for the test bench can be developed using the Synopsys Graphical Environment[6] to interconnect test bench primitives taken from a library. These library primitives are also developed using CASE tools, e.g., Express VHDL and Cadence (Comdisco) SPW[5].

The system has an intelligent interface that prompts the user for high level test bench information that is used to produce data for simulating the model under test. Typically, the input information keyed in by the user may be the initial parameters about a target, e.g., position, velocity and direction of motion or technical characteristics of a radar that is used for processing signals. These parameters can also be fed in as data files. The user selects between behavioral and structural models of the test bench for simulation. The simulation results are fed to the model under test. The comparator compares the model response with the expected response.

5. SIMULATION OF TEST BENCH MODEL

The test bench model has to be simulated to provide the required stimuli for the model under test. Based on the input information obtained by the intelligent interface, a simulation control file is created, that is used to simulate the test bench model (either behavioral or

[2] We are reusing the word environment in a more restricted sense here.

structural). Depending on this file, the inputs to the test bench can be either user-fed or obtained from a data file.

6. CONCLUSION

In this paper we have presented approaches to the high level generation of testbenches for VHDL signal processing models. The methodology under development provides a structured approach which: 1)relieves the modeler of the details of testbench development, and 2)significantly speeds up this development.

7. REFERENCES

[1] C. H. Cho and J.R. Armstrong, "The B Algorithm: A Behavioral Test Generation Algorithm", Proceedings of 1994 International Test Conference, pp. 968-979.

[2] S. Kapoor, J. R. Armstrong, and S. R. Rao, "An Automatic Test Bench Generation System", Proceedings of the VHDL International Users Forum, Spring 1994, PP 8-17.

[3] G. A. Frank, "The Evolution of External Models For Simulation", Proceedings of the Systems Design Synthesis Technology Workshop, Naval Surface Warfare Center, Silver Spring, MD Sept. 10-13, 1991

[4] Express V-HDL - User Reference Manual, Volume I, Version 3.0, December 1993

[5] Comdisco Systems - SPW - The DSP Framework, Designer/BDE User's Guide, March 1994

[6] Synopsys Graphical Environment User Guide, Version 3.0, December 1992

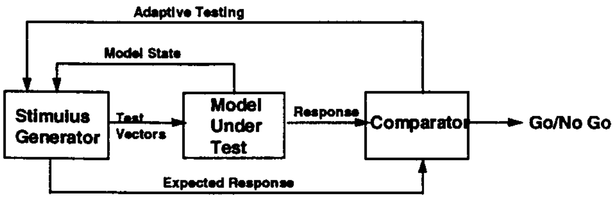

FIGURE 1. A Basic Test Bench

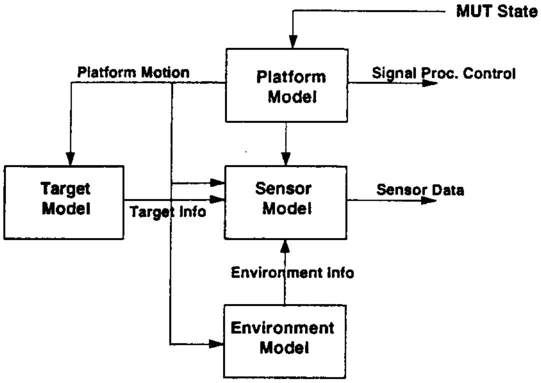

FIGURE 2. RASSP Stimulus Generator

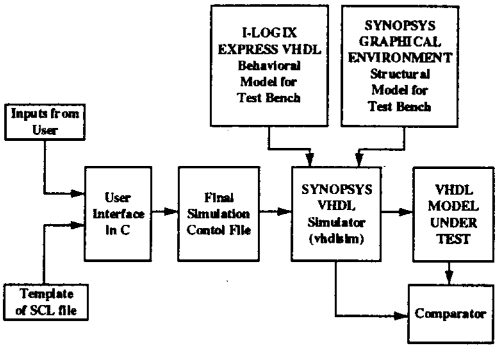
I-LOGIX EXPRESS VHDL Behavioral Model for Test Bench
SYNOPSYS GRAPHICAL ENVIRONMENT Structural Model for Test Bench
Inputs from User
User Interface In C
Final Simulation Contol File
SYNOPSYS VHDL Simulator (vhdlsim)
VHDL MODEL UNDER TEST
Template of SCL file
Comparator

FIGURE 3. A System for Test Bench Development

,