HDL Bencher XE for Fast Behavioral FPGA Verification

Now you can develop complete, timing constrained VHDL and Verilog testbenches in minutes.

by Andy Bloom
Director of Engineering, Visual Software Solutions
(info@testbench.com)
ambloom@testbench.com

Ricky Escoto
Director of Marketing, Visual Software Solutions
(info@testbench.com)
rescoto@testbench.com

Validating FPGAs can require substantial effort, unless you have a high order software tool like HDL Bencher XE. The usual process requires you to write many testbenches, simulate them, check the results, and log all failures. To adequately test a design involves verifying all the possible cycle types available to the device, and may require several hundred test cases. And, as your design is revised, port definitions may change, making your existing testbenches obsolete, which results in unnecessary effort to update the HDL source code and the accompanying testbench.

To simplify FPGA and CPLD testing, Xilinx now includes Visual Software Solutions' HDL Bencher XE in the Foundation ISE and WebPack ISE design tools. No knowledge of HDL or scripting is required.

HDL Bencher Overview

HDL Bencher accepts any HDL design, and then lets you select the unit under test (UUT), specify stimulus and response (using the pattern wizard and the WaveTable™ spreadsheet-based interface), and then export a complete, self-checking testbench automatically. If your HDL source contains external dependencies, HDL Bencher prompts you to compile them locally so that the whole design can be simulated.

HDL Bencher lets you manipulate waveforms in the same way you manipulate a spreadsheet. You can cut, insert, and paste rows (signals) or columns (time regions) with ease, and HDL Bencher automatically readjusts timing.

Interactive Simulation

The testbenches are automatically updated when the HDL source changes, eliminating stale test cases. To facilitate design retargeting, HDL Bencher allows testbenches to be moved between VHDL and Verilog with one simple command. When compilation errors are found during simulation, HDL Bencher links the error reported to the offending line in the HDL source.

Create Self-Checking Testbenches

The testbenches include component instantiations, generic specifications, stimulus, output check procedures, and assertions. You can create "golden models" for regression testing and future design validation; mismatches in expected and actual output values are flagged automatically. All the necessary timing constraints are faithfully represented in the resulting testbench.

Verify Timing

By adding timing constraints, you can generate VHDL or Verilog testbenches for post-synthesis verification. Synthesized netlists differ from behavioral HDL because data types are remapped, I/O modes are changed, unused signals are dropped, and generics are flattened. HDL Bencher automatically remaps behavioral testbenches to simulate with synthesized netlists.

Demonstration Design

As an example, the following HDL code is used as input into HDL Bencher:

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
entity counter is
    Port (
      CLK,RESET,CE : in std_logic;
      T : out std_logic;
      COUNT : inout integer range 0 to 7 := 0);
end counter;
architecture behavioral of counter is
begin
   process (CLK, RESET, CE, COUNT)
   begin
      if RESET=’1’ then
         COUNT <= 0;
      elsif CLK=’1’ and CLK’event then
         if CE=’1’ then COUNT <= COUNT + 1;
         else COUNT <= COUNT;
         end if;
      end if;
      if COUNT=1 then T<=’1’; else T<=’0’; end if;
   end process;
end behavioral;
```

Within the Xilinx ISE software, you start by selecting the HDL file from the source window, then you choose HDL Bencher from the process window. The design is automatically imported, and you are given the opportunity to select worst-case global timing parameters:

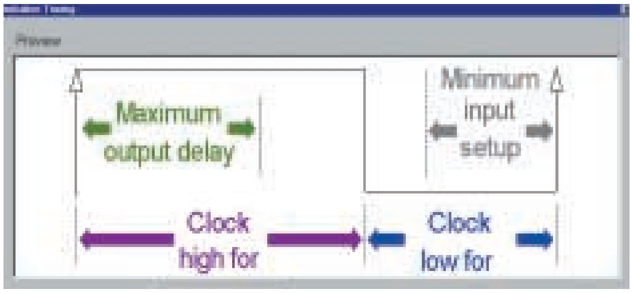

Figure 1 - Initial timing dialog

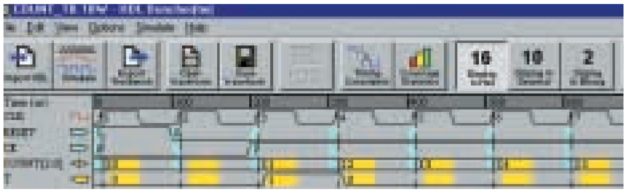

Figure 2 - Stimulus for the design example

Figure 3 - ModelSim running the testbench and design

A waveform is created next (Figure 1), which includes all the signals for the unit under test (UUT). Individual waveforms are then modified directly on the screen by clicking on the signals to show the expected behavior, or by using the built-in pattern generator.

Next, HDL Bencher automatically exports a self-checking testbench. The testbench includes all stimulus (Figure 2), output assertions, timing constraints, and check routines needed to verify the operation of the design. The testbench is added to the ISE project, then auto-simulated through the Xilinx ISE software and ModelSim (Figure 3).

An advanced version of HDL Bencher is now available which automatically back-annotates the expected response into the waveform. If no expected response was specified, HDL Bencher back annotates the response obtained by ModelSim. Otherwise, expected and actual responses are compared, and discrepancies are highlighted.

Once your design is synthesized, its behavioral testbench may be incompatible with the resulting VHDL netlist generated during the post-route process. In this case, the resulting netlist uses std_logic_vector instead of integers. To make the synthesized netlist simulate, you would switch back to HDL Bencher, re-associate the waveform with the synthesized netlist, and re-export the testbench. Finally you would switch back to the ISE software and re-simulate.

The Resulting Testbench

The exported testbench in this example is 183 lines of code, and took under 1 minute to create and simulate. The following portions of the testbench highlight some of the aspects of automatic testbench generation:

```
Automatically Commented
      — VHDL TestBench created by
      — Visual Software Solution’s HDL
        Bencher 2.00
Libraries Extracted
    LIBRARY IEEE;
    USE IEEE.std_logic_1164.all;
Log File Created
    FILE RESULTS: TEXT IS OUT
“results.txt”;
Components Instantiated
    COMPONENT counter
      PORT (
        CLK : in std_logic;
          :
          COUNT : inout integer RANGE 0 TO 7
Test Signals Defined
    SIGNAL CLK : std_logic;
          :
    SIGNAL COUNT : integer RANGE 0 TO 7;
Instantiates Unit Under Test
    UUT : counter  PORT MAP (
      CLK => CLK,
          :
      COUNT => COUNT
Clock Process Created
    BEGIN
      CLOCK_LOOP : LOOP
      CLK <= transport ‘0’;
      WAIT FOR 10 ns;
      CLK <= transport ‘1’;
Creates Check Procedures
    PROCEDURE CHECK_COUNT(
        NEXT_COUNT : INTEGER
Reports Errors In Expected Values
      IF (COUNT /= NEXT_COUNT) THEN
        write(TX_LOC,string’(“Error at
time=”));
Applies Input Stimulus
      RESET <= transport ‘1’;
      CE <= transport ‘0’;
Validates Timing
      WAIT FOR 100 ns; — Time=820 ns
Verifies Outputs
      CHECK_COUNT(7,820); — 7
Reports Success/Failure
    ASSERT (FALSE) REPORT
          “Simulation successful. No prob-
          lems detected. “
Draw Expected Behavior
```

Conclusion

With HDL Bencher you can verify the operation of VHDL and Verilog designs in minutes; no HDL scripting is needed. The resulting testbenches are self-checking, and are compatible with the Xilinx ISE software. HDL Bencher XE is available at no charge to all Xilinx customers, and is included with the ISE software or can be downloaded from www.xilinx.com (download the HDL Bencher “BackPack” from the WebPack section).