We take you to the leaders.

HDL VERIFICATION SPECIAL SECTION

by Michael A. Bohm,
Exemplar Logic,
VP, Chief Scientist,
bohm@exemplar.com

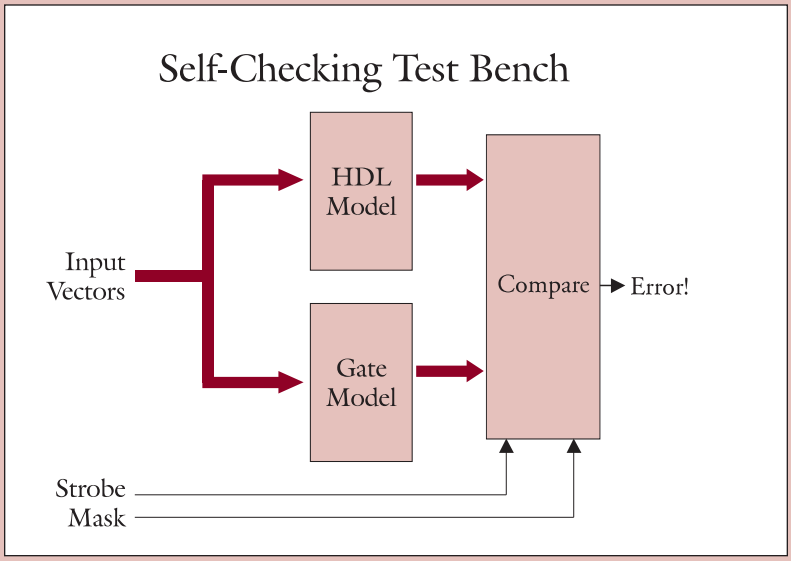
Verification Using a Self-checking Test Bench

As an FPGA designer, your life is basically one big debug cycle. From the moment you receive the first specification from marketing, until production silicon is ready, you are looking for problems.

Once you recognize a problem, then corrective action must be identified. This process of problem identification and correction is the slowest part of the design process, mainly due to the "human factor." Your ability to take in all available information, evaluate the data, and come to a conclusion is one of the main bottlenecks in producing complex FPGAs.

In the days when designs stayed below the 10K-gate barrier, you were capable of handling designs using schematic capture. The amount of design information that you had to control was manageable, with low risk. As designs got larger, Hardware Description Languages (HDL) allowed you to describe an FPGA in a more abstract and compact form. This new form, along with functional simulation, allowed you to define, understand, and build very complex FPGAs.

The HDL developed for an FPGA can be thought of as an executable specification for the device being designed. The problem with this specification is that it only contains the functional information. The AC, DC, and physical information about an FPGA come from the transformation performed during logic synthesis and place and route.



With all the changes that occur from transforming technology-independent HDL to a technology-dependent FPGA device, it is a good design practice to perform a final verification stage before creating silicon. This final verification will prove that the original HDL performs to specification and minimizes the risk in the final device.

One way to perform this verification is through a self-checking test bench. This test harness will not only prove design correctness, but will also provide a structure that is simple and easy to debug. The diagram below illustrates this principle.

The self-checking test bench has three main blocks: the original HDL model, the final gate-level model, and a comparison block. The test bench works by comparing the results from the two FPGA models, which both receive the same input stimulus, and then flags any discrepancies that are found. The real power in this method is that when an error is found, you can probe into both models and trace signals to see what is causing the problem.

The comparison block provides the brains in a testbench. This block performs a logical compare of the primary results of both models. The "strobe" pin on this block decides when data should be valid in both models. This is usually just before the end of the clock period. The "mask" pin tells the comparison block when to ignore data from the models. This is used during the initialization phase of the simulation. Usually the HDL model is in a known state from the beginning of simulation, while the gate level simulation will take a few clock cycles to settle down. The other feature of the comparison block is that it can stop the simulator when an error occurs by executing a VHDL "assert." This is nice feature because the simulator has stopped at the exact point of an error so you can then debug the design.

Conclusion

A self-checking test bench is a great method for performing that final verification stage before creating silicon. It gives you the extra confidence that the design will be correct and smoothly roll into production. ◆