Random Code Generation

The complexity of processor designs has increased dramatically in an effort to improve performance, reduce system cost, and allow processors to be used in more system configurations. The increasing complexity makes it almost impossible to identify the specific event cross-products that need to be tested to ensure that a design is correct. Random code generation is an effective method for testing a design without having to identify exactly what needs to be tested. A random code generator creates legal, random sequences of machine states and instructions that exercise a design more thoroughly than application software.

The term random is somewhat misleading—generating completely random machine states and instructions would result in uninteresting tests as far as stressing the design is concerned. Instead, generators focus on key aspects of the design while preserving an element of randomness. Accelerating rare events, hitting boundary conditions, and concentrating on instructions that exercise complex parts of the design are among the ways to focus a generator. The probabilistic distribution of random numbers creates interesting combinations of these focused events.

Although random code generation has higher coverage in postsilicon testing where the design can be tested at high speeds, it can also be effective in presilicon testing. When running on relatively slow presilicon models, the effectiveness can be improved by adding more elaborate checking strategies and focusing the generators on smaller portions of the design.

Some elements of a quality random code generator include:

- Coverage of the entire design
- Focus on complex portions of the design
- Low fault latency (i.e., a failure gets noticed soon after it occurs)
- Reproducible test cases
- Aids for debugging failing tests.

Random testing techniques can also be applied to designs other than microprocessors. Memory or I/O controllers can use these techniques to randomly generate machine state and transactions that will stress the controllers. Designing special-purpose bus exercisers that are controlled by random test generators can extend such testing into the postsilicon environment.

▶ Return to Article 9

▶ Go to Next Article

▶ Go to Journal Home Page