Constructing test sequences for hardware designs with parallel starting operations using implicit FSM models

Mikhail Chupilko
Institute for System Programming of the Russian Academy of Science
DUT Black Box Representation

Affecting the inputs

Design Under Test

Checking the outputs

Stimuli

Test System

Reactions
Operations, Micro-Operations, Etc

CLK

Stage 0
stimulus

Stage 1
reaction

...  

Stage N
reaction
Using FSM to Testing

- We propose using FSM as it allows stimuli creation in automatic way based on specifications in a form of pre- and post-conditions
  - Pre-conditions mean allowing to start operations
  - Post-conditions should be checked when operations is done.
Problem of Multi-Stimuli Creation Statement

- Describing all possible combinations of parallel starting operations in case of complex compositional FSM has some disadvantages:
  - It has a redundant code which is changed every time when compatibility of parallel operations changes
  - It is hardly configurable
Basic Way of Describing Current FSM State

\[\{(A, 1)\} \cup\]
\[\text{next}(S, A) = \{(B_i, N_i) | \text{pre}(B_i, N_i) = \text{false}\} \cup\]
\[\{(B_i, N_i + 1) | \text{pre}(B_i, N_i) = \text{true} \land N_i < N\}\]
Modified Way of Describing Next FSM State

Operation A is replaced by \((A_1, A_2, \ldots, A_K)\) as a one pool of wires divided into some domains.

\[
\{(A_1, 1), (A_2, 1), \ldots, (A_K, 1)\} \cup \\
\text{next}(S, A) = \{(A_i, N_i) | \text{pre}(A_i, N_i) = \text{false}\} \cup \\
\{(A_i, N_i + 1) | \text{pre}(A_i, N_i) = \text{true} \land N_i < N\}
\]
Constructing Multi-Stimuli

Stimuli from the test engine

Buffer accumulator for stimuli

To the inputs of the DUT

CLK

A   A   A, B
A   A   A, B
A    A, B

Stimuli
Buffer
Started operations
Example of FSM

Operations were sent, New state - []
Case Study – L2 Cache of MIPS64-Comp. Microprocessor

- Reducing code of test system more than to two times: from 14 CLOK to 7 CLOK
- Found domains: $D_1 = \{\text{Load-Data, Stores, Caches}\}$, $D_2 = \{\text{Load-Instruction}\}$, $D_3 = \{\text{Snoops}\}$
Future Work

- Improvement of tool support for specification and tests development
- Integration with the modern Open Verification Methodology (OVM)
Contacts

- Institute for System Programming of RAS
  http://www.ispras.ru

- UniTESK Technology
  http://www.unitesk.com

- Hardware Verification at ISPRAS
  http://hardware.ispras.ru

- Mikhail Chupilko
  chupilko@ispras.ru
Thank You!

Questions?