Community Based Verification of
MIPS Microprocessor using UVM

Prof. Morris Jones, Saurabh Purohit, Udit Jaisalmeria
Department of Electrical Engineering, San Jose State University, San Jose, California 95192

Introduction
Our project focuses mainly on the verification methodology to test a particular design. We picked out ‘MIPS Microprocessor Design’ as our targeted design. We aim to create a UVM verification
environment that is completely scalable to accommodate large number of identical designs and then perform a clean verification that compares the responses from all the designs. We call this verification as Community Based Verification[1].

Key Features of UVM:
• It has SystemVerilog Base Class Library.
• It is based on constrained Random Verification approach
• Powerful stimulus generation mechanism using sequences
• It supports reuse
• UVM configuration helps in customizing the environment from top or anywhere[2]
• UVM configuration helps in customizing the environment from top or anywhere[2]

Verification Approach
Following are the basic building blocks of our UVM framework:
- Transaction Item (mips_sequence) is a packet full of information which we pass through the framework.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Opcode</th>
<th>in</th>
<th>op</th>
<th>out</th>
<th>inst</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD</td>
<td>001000</td>
<td>NOL</td>
<td>NOL</td>
<td>NOL</td>
<td>NOL</td>
</tr>
<tr>
<td>SUB</td>
<td>001001</td>
<td>NOL</td>
<td>NOL</td>
<td>NOL</td>
<td>NOL</td>
</tr>
<tr>
<td>ORI</td>
<td>001010</td>
<td>NOL</td>
<td>NOL</td>
<td>NOL</td>
<td>NOL</td>
</tr>
<tr>
<td>NOR</td>
<td>001011</td>
<td>NOL</td>
<td>NOL</td>
<td>NOL</td>
<td>NOL</td>
</tr>
</tbody>
</table>

The most challenging and interesting part of UVM based verification is to create a way to build three different types of sequences (R-type, I-type, J-type). We adopted an approach in which we use a single transaction item names ‘mips_sequence’ which has all the fields required by the three types of instruction formats. Table shows the contents of our transaction item

The similarity in the responses from the two DUTs is measured from these parameters:

Key References

Results & Analysis
Design Inconsistency: We found that the output responses for the random tests were not identical. Results shown in table below:

<table>
<thead>
<tr>
<th>DUT</th>
<th>DUT1</th>
<th>DUT2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>SUB</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>ORI</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>NOR</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

Stimulus generation (green signals)

Output response waveform

Conclusions
The two designs were compared by injecting same random transactions multiple number of times. The results shows that there are some loopholes in one or probably both the designs. But as per the objective of our project, the result of our observation of responses is that we were able to find the test cases for which the two designs disagreed. As a result, it becomes important for the designers of the DUTs to review their designs again to sort out the differences. This project can be extended to include any number of designs and rather than building a UVM checker and a predictor, we can simply use the results from the cloud of designs and know if the design under consideration is faulty.

Acknowledgments
We would like to thank Prof. Morris Jones and Dr. Thuy Le for giving us the opportunity. Prof. Morris Jones’s reviews, comments and discussions on this project work were really valuable.

For further information
Please contact udit.jaisalmeria@sjsu.edu or saurabh.purohit@sjsu.edu.