SystemC Manipulation Framework: from RTL VHDL to Optimized TLM SystemC

Syed Saif Abrar, Valentin Tihomirov, Maksim Jenihhin, Jaan Raik
Tallinn University of Technology, ESTONIA
{ saif \ valentin \ maksim \ jaan }@ttu.ee

Abstract

We propose a novel framework for SystemC manipulation based on the open-source hardware design and analysis environment zamiaCAD. The framework provides optimized VHDL-to-SystemC translation and subsequent abstraction to higher-level, including an Eclipse-based front-end.

1. Overview of zamiaCAD

zamiaCAD [1] is a modular and extensible open source framework supporting multiple use-cases, like hardware design and debug, along with conversion and abstraction to SystemC, conforming to the IEEE Standard VHDL Language Reference Manual. Fig.1 shows the structure and applications of zamiaCAD. It can handle large industrial designs, such as a SoC made of 3500 Leon3MP-s, and provides all facilities through Eclipse-based GUI front-end.

2. VHDL to SystemC Translation

IP design-houses have a large number of legacy IP cores described in VHDL whose reuse and integration into SystemC ecosystem is highly demanded. The zamiaCAD framework includes open-source methodology [2] to convert RTL VHDL IP cores to cycle-accurate SystemC designs. The SystemC output is emphasized to be human-readable and provides clear correspondence to the source VHDL code, thus allowing full manual code changes and debug.

3. SystemC Optimizations

Various optimizations are performed on the SystemC code generated by zamiaCAD, such as: (1) Using native C++ data-types for faster simulation, (2) Optimization of VHDL concurrent statements, (3) Abstraction for co-simulation.

4. SystemC Abstractions

1) Interface abstraction deals with the conversion of signal-level and cycle-accurate protocol to the standard TLM-2.0 protocol is SystemC. Our methodology can be used to abstract any arbitrary signal-level protocol to TLM-2.0, without affecting side-band signals like clock and interrupts.

2) Clock abstraction aims at removing the clock-signal, either partially or completely. Minimization of delta-cycles and events in a system-level model is the underlying idea.

3) Functional abstraction focus on abstracting the behavioral implementation of a module. Notion of SystemC-based Loose Modeling (SCLM) is introduced to functionally abstract a VHDL FSMD design, using goto statements. SCLM enables to neglect design model parts irrelevant for a particular abstraction. SCLM goto models can be easily transformed to a readable structured form (e.g. WHILE, FOR, etc). Table-I presents the experimental results for simulation speed-up provided by different stages of abstraction, for various benchmark designs. Un-optimized goto-code (GT) provides for a small speed-up. The optimized goto-implementation (GT_O) improves the simulation speed a little more. The last column demonstrates speed-up achieved by the untimed optimized goto (GT_O_U) model. Here the data dominated designs with less interruption of the computational phase by communication such as b07 and codec demonstrate a significant speedup up to 27.24 times.

<table>
<thead>
<tr>
<th>Design</th>
<th>RTL</th>
<th>FSMD</th>
<th>GT</th>
<th>GT_O</th>
<th>GT_O_U</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gcd</td>
<td>1</td>
<td>1.25</td>
<td>1.71</td>
<td>2.65</td>
<td></td>
</tr>
<tr>
<td>b02</td>
<td>1</td>
<td>1.03</td>
<td>1.03</td>
<td>1.94</td>
<td></td>
</tr>
<tr>
<td>b07</td>
<td>1</td>
<td>1.09</td>
<td>1.10</td>
<td>12.08</td>
<td></td>
</tr>
<tr>
<td>Codec</td>
<td>1</td>
<td>1.35</td>
<td>1.42</td>
<td>27.24</td>
<td></td>
</tr>
</tbody>
</table>

Acknowledgements

The authors would like to thank Guenter Bartsch, the main developer and project founder of zamiaCAD, for the source code and the help throughout the development and debug. The work has been supported in part by Estonian Science Foundation grant 9429, EU through European Regional Development Fund.

References

[1] zamiaCAD webpage: http://zamiacad.sf.net/