Contribution ID: **98** Type: **Poster**

Easy and structured approach for software and firmware co-simulation for bus centric designs

Tuesday, 21 September 2021 17:26 (3 minutes)

Although software and firmware co-simulation is gaining popularity, it is still not widely used in the FPGA designs.
This work presents easy and structured approach for software and firmware co-simulation for bus centric designs.
The proposed approach is very modular and software language agnostic.
The only requirement is that the firmware design is accessible via some kind of bus.
The concept has been used for testing DAQ system being developed for high energy physics experiment.

Summary (500 words)

Software and firmware co-simulation can save a lot of time and money, as it leads to lower number of HDL project builds and reduces the number of test iterations with the real hardware.
Despite its obvious advantages the co-simulation is still rather rare to see in FPGA designs.
There are at least four reasons for such situation.
The first one is that setting up a co-simulation framework requires knowledge on multiple computing areas.
The second one is that it might be time consuming.
The third one is that ready to use frameworks sometimes do not support some of the desired HDL features, for example handling compound types such as records or arrays.
The fourth one is that ready to use frameworks are strictly coupled with a single programming language.
This work presents modular approach and tries to be inline with the Unix philosophy.
The framework blocks are loosely coupled and each of them can be easily replaced.
The co-simulation framework consists of the following elements: software co-simulation interface, HDL co-simulation interface, HDL BFM (Bus Functional Model), test runner.
The whole test bench additionally consists of the projects software and firmware code.
The co-simulation interfaces are relatively simple and short, and once written they can be reused for different tests within the project.
If different software languages are used for the prototype and target implementation phases it is also easy to write a co-simulation interface for the new language and reuse the co-simulation framework from the prototyping phase.
The BFM can be custom or taken from a library such as UVVM.
The idea is based on the assumption, that all communication is done via the bus.
Not only the regular data is transferred via the bus, but also the test bench specific data.
Such approach is immune to lack of support for compound types.
The proposed approach is not free of drawbacks.
The first one is that the firmware design must have some kind of bus.
This should not be a problem as almost all complex FPGA designs have some kind of bus, Wishbone and AXI being probably the most popular.
The second one is that doing precise timing checking between signals is hard to achieve solely within test bench software.
It requires firmware checker accessible via the bus.
Another approach is using PSL (Property Specification Language) or SVA (SystemVerilog Assertions).
The proposed approach has been used for testing of DAQ (Data Acquisition) system for the CBM (Compressed Baryonic Matter) experiment that is being prepared at FAIR (Facility for Antiproton and Ion Research) in Darmstadt.

Primary author: KRUSZEWSKI, Michal (Warsaw University of Technology)

Presenter: KRUSZEWSKI, Michal (Warsaw University of Technology)

Session Classification: Posters Programmable Logic, Design Tools and Methods

Track Classification: Programmable Logic, Design Tools and Methods