SystemC Evolution Day
Wrap-up and Next Steps
Technical sessions

• SystemC/TLM and Language Standards (C++11/14)
• TLM Standard for Serial Interfaces
• Multi-Threaded SystemC and External Interfaces
• UVM-SystemC & Config., Control and Inspection
Using Modern C++ in SystemC Modeling
- Significant productivity gain seen in modeling domain

C++ standard in IEEE 1666-20XX
- Base next version of standard on C++14
- Compatibility not seen too critical, as standard moves slowly anyway

C++14 SystemC API extensions should focus on enabling designer productivity
- Detailed feature discussions in extended minutes
- Incremental addition of new features, without immediate compatibility break on old compilers
TLM Standard for Serial Interfaces

• Summary
  – ~20 attendees
  – Several approaches
    • TLM-2 using OSI layers to define abstraction levels.
      – Having a blueprint that should be followed by each protocol implementation.
      – Minimize GP, use extensions for protocol specific details.
    • Bottom-up – implement dedicated protocols without the need for a generic umbrella.
  – Many questions e.g.
    • Router model needed required for all protocols?
    • Which limitations are acceptable?
    • Different abstraction level due to orthogonal use cases.
      – Multiple timing points

• Next Steps
  – Address questions in TLMWG
  – Source code examples to discuss more concrete issues
Multi-Threaded SystemC and External Interfaces

• Two Views on Parallelism
  – Fine-grained automatic
    • Not really discussed in the session
  – Coarse-grained manual
    • Controlled by the modeler

• The proposed synch mechanism is a good thing and should be moved on to the standards working group

• Exact rules for how to sync simulator time varies with the use case, no consistent picture emerged in the discussions
Two Views on Parallelism

• Fine-grained automatic
  – Not really discussed in the session

• Coarse-grained manual
  – Controlled by the modeler
  – The proposed synch mechanism is a good thing and should be moved on to the standards working group

• Exact rules for how to sync simulator time varies with the use case, no consistent picture emerged in the discussions
Coarse-Grained Manual

Each proxy has to be able to wait independently: proposed async wait

SystemC model
proxy
SystemC model
proxy
SystemC model
proxy
SystemC kernel
SystemC kernel
SystemC kernel
Coarse-Grained Manual II

Manually set up threads on top of a kernel, implement thread safety in some way. Implement communication in a thread-safe way.
Trade-Off

Fast

Understandable

Safe
UVM-SystemC & Config., Control and Inspection

• SystemC Configuration Use Models
  – Combine or separate UVM-SC and CCI approach
  – Use model: Tool/model interface and/or tool/testbench interface

• Register interface
  – uvm_reg versus sci_reg: combine or separate
  – Use model: connect UVM-SC backdoor to a HDL or SystemC objection via the HDL path
Main Points (Configuration)

- UVM primarily aimed at testbench (e.g. blackbox testing)
  - Type agnostic, simple set & get, global database, DUT-non-intrusive
- CCI primarily aimed at IP model configuration
  - More sophisticated (e.g. broker layer, access controls, ...)
- Are both configuration developments too far to cancel one of them?
  - Can one be based on the other?
  - Both access philosophies needed to fulfil requirements
- Where/how should seed values be configured? (e.g. variation simulation)
  - Some CCI-parameter control from UVM-SC may be necessary
Main Points (Register interface)

• Register API available from Cadence & ST
  – Layer between model and tool
Next Steps

• Access of DUT configuration from the UVM-Testbench (CCI)

• Features/Function Matrix of UVM-SC and CCI config methods/APIs needed to prove if implementation can be based on each other (CCI&VWG)
Join A Working Group And Contribute!

SystemC Synthesis Working Group (SWG)

Charter
This group is responsible for the definition of a synthesizable subset of SystemC.
Chair: Andres Takach, Mentor Graphics
Vice-Chair: Michael Meredith, Cadence

Background
Since the last public review of the Synthesis Subset (version 1.3 which was released in August 2009), draft 1.4 has been updated to improve the clarity of what constructs are supported. The draft was reviewed and updated based on the IEEE 1666-2011 SystemC language standard. Updates were done in all sections including support for C++ constructs and SystemC modules, processes, clocks, resets and datatypes.

The SystemC Synthesizable Subset Version 1.4 was open for public review and comment until July 13, 2015. Community feedback on the draft is now being considered for the next release of this standard. Although the period to submit comments has ended, the preview document can be downloaded here. The release of version 1.4 will be announced soon.

Join this Working Group
If you are an employee of a member company and would like to join this working group, click here (requires login) and click Join Group. WG participation requires right of entry by the group chair.
SystemC Community

- Online at [http://accellera.org/community/systemc](http://accellera.org/community/systemc)
- Community forums, upload area for contributions, SystemC news
Advancing Standards Together

• Share your experiences
  – Visit www.accellera.org and register to post on community forums at forums.accellera.org

• Show your support
  – Record your adoption of standards

• Become an Accellera member
  – Join working groups

• Join face-to-face meetings
Dinner

- Taj am Campeon – [http://www.taj-am-campeon.de](http://www.taj-am-campeon.de)
- 19:30 – 21:30
- Building 10 – here on campus

- Please announce your participation until 11:00!
- (Food/drinks not included – self-paid)
Thank you!

Accellera Global Sponsors

© Accellera Systems Initiative