SystemC 2.1 Overview

OSCI Language Working Group

February 2004

Goals of SystemC 2.1

- Improve modularity for IP delivery
  - As SystemC 2.0.1 was more widely used for transaction-level design IP and verification IP, several practical barriers emerged
- Provide better support for transaction-level design and verification
  - Requirements identified for TLM prior to TLM WG
  - Requirements identified by Verification WG
- Resolve longstanding ease-of-use issues
  - Inconsistency, non-orthogonality, platform support
- Fix bugs
  - Of course

Modularity, IP Delivery Capability

- Structured error reporting mechanism
  - Provides consistent messaging from
    - The core simulator
    - Add-on libraries
    - IP modules
    - Testbenches
  - Can be customized by vendors for integration with co-simulation messaging
- Access to start-up arguments
  - sc_argc() and sc_argv() give access to argc and argv from outside of sc_main
  - Allows command line control of Design IP and Verification IP
  - Allows command line control of add-on libraries
- New callbacks allow IP integration without needing code in sc_main
  - before_end_of_elaboration()
  - start_of_simulation()
  - end_of_simulation()

Transaction-level Design and Verification

- sc_export
  - Provides a modular capability for a module to advertise internal interfaces for access from outside
- Dynamic process support
  - Crucial for development of transaction-level testbenches
  - Also important for software modeling
  - Thread creation example from 2.0.1 has been enhanced and incorporated into the language
  - Kernel automatically allocates and reclaims threads as needed. No “thread pool” required.
  - Uses publicly available boost::bind library
- sc_event_queue class
  - Catch multiple calls to notify() in same delta cycle
  - Allows IP and testbenches to reliably catch every notify()

Resolve Longstanding Ease-of-use Issues

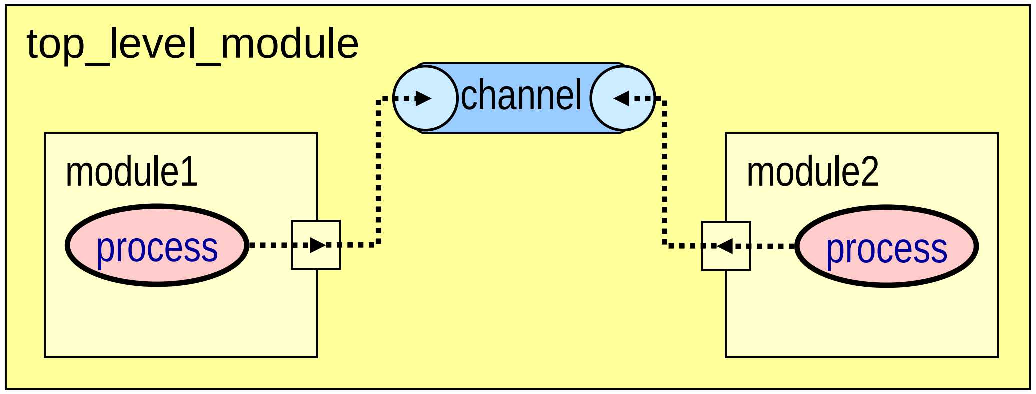
- Support for programs with their own main() function
  - Programs with their own main() function can call sc_main_main() to perform SystemC processing.
- Mixed concatenation
  - Concatenations of sc_uint, sc_biguint, sc_int, sc_bigint, sc_signed, sc_unsigned etc can now be mixed without ugly casting
  - Concatenations can produce results greater than 64 bits
- New API for obtaining process kind info: sc_get_cur_process_kind()
- Object code release tagging
  - link-time detection between incompatible object files
- POSIX thread support
  - Allows use of memory leak checking tools
- Support for MacOS X

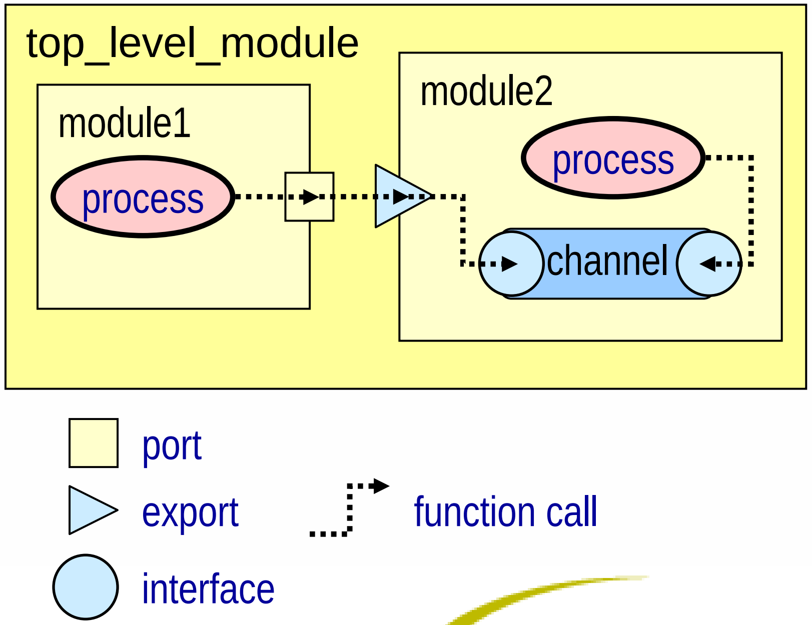
Importance of sc_export

- sc_ports facilitate modular design by precisely declaring interfaces *required* at a module boundary
- sc_exports facilitate modular design by precisely declaring interfaces *provided* at a module boundary
- sc_ports and sc_exports allow interfaces to be passed through each level of the hierarchy
- Use of sc_port and sc_export improves modularity by avoiding reliance on explicit multilevel paths
- sc_export permits direct function call interfaces for TLM without introduction of extra process switches

Technical Details of sc_export

- SystemC 2.0
  - sc_port indicates sc_interface is *required* by an sc_module
  - sc_interface implemented by channel higher in the hierarchy
  - sc_module calls sc_interface function through sc_port
- SystemC 2.1
  - sc_port mechanism exists unchanged
  - sc_export indicates sc_interface is *provided* by an sc_module
  - sc_interface implemented somewhere within the sc_module
  - sc_port can be bound directly to sc_export
  - Other sc_module calls sc_interface function through sc_port and sc_export





Bug Fixes

- sc_start() at max value aborts simulator
- sc_trace for uint64, int64 missing
- sc_set_time_resolution not properly affecting VCD dump information.
- The value of sc_clock needs to be updated during update phase, not execution phase.
- sc_string subscript operator may modify multiple instance because of copy semantics.
- Cpu risc example not shipped anymore
- Error in sc_bv char constructor
- sc_biguint partial selection bug
- Missing terminating null char in >> operator for sc_string.
- The constructor sc_module(const sc_module&) is not defined
- Signal initialized in module CTOR not registered with module.
- Deletion of main fiber should not occur in ~sc_cor_fiber
- Need ability to compile with Wno-deprecated
- Tracing ports after end_of_elaboration had no effect
- wait() in module ctor led to crashes.