SystemVerilog C/C++ Committee

LRM Status as of 2/28/03

Swapnajit Mittra
Chair
SGI

Ghassan Khoory
Co-Chair
Synopsys

SV-CC

- Focus Areas
  - DirectC API (considering official name of DPI)
  - Foreign Code Inclusion
  - Assertion API (as VPI extensions)
  - Coverage API (as VPI extensions)
- Current Status (for all three)
  - All major technical discussions & decisions have been completed
  - All APIs have gone through two draft revisions in the SV 3.1 LRM format
  - Coverage and Assertions parts are ready for review by other committees

SV-CC

- Focus for Short Term
  - Close on all outstanding inter-committee issues
    - Extern Proposal
  - Continue review of draft LRM by committee
  - Decide on issues identified for consideration in SV 3.2+
    - Expansion of VPI to address SV 3.1 types & new constructs
    - Handling references & classes across the “DPI”
    - Foreign source inclusion & unified VPI linking model
    - Additional coverage metrics support
  - Review BNF and LRM for consistency with other SV 3.1 committees

SV-CC LRM Overview

- Direct Programming Interface, DPI - SV Layer
  - Two layers specified – SV and foreign language
  - Direct function calls use native syntax and semantics
  - All SV types are supported across DPI (with minor restrictions)
  - Only 0-time functions can be invoked
  - DPI functions can be classified as pure or context
    - Pure functions must not have side effects
    - Context functions are aware of their invocation instance and may use other interfaces, ie. PLI, VPI
  - Memory owned by SV and C are disjoint

SV-CC LRM Overview

- Direct Programming Interface, DPI - C Layer
  - Two header files provided, svc.h & svc_src.h
  - Two levels of application portability, binary and source
  - Data representation across DPI
    - Simple SV types transformed appropriately to C types
    - Complex SV types passed by reference
  - SV Array ranges are normalized across DPI for compatibility with C
  - Encoding of 4 valued logic compatible with VPI
  - C code can invoke exported SV functions
  - C pointers can be passed to SV handles and vice versa

SV-CC LRM Overview

- Foreign Code Inclusion
  - Standard mechanism for associating foreign code with SV
    - Location of files
    - List of files to be loaded
    - Bootstrap files associated with application
  - Recommended, but not mandated, CMD line switches
    - Semantics are defined, implementation are free to define switch naming convention

SV-CC LRM Overview

- Assertion API – VPI Extensions
  - Getting handles for assertions
    - By name
    - Iterate over all assertions in an instance or design
  - Obtaining assertion info from a handle
    - Source info; line, file, etc.
    - Name, instance, module, clocking expr, and directive
  - Putting callbacks on assertions
    - Start, Success, Failure, Step, Enable, Disable, Reset, Kill
  - Controlling assertions
    - Enable, Disable, Reset, Kill, Enable/Disable Stepping

SV-CC LRM Overview

- Coverage API – SV Side
  - Definitions for Coverage Types
    - Statement, toggle, FSM State, Assertion
  - Five system tasks defined
    - $coverage_control
    - $coverage_get_max
    - $coverage_get
    - $coverage_merge
    - $coverage_save
  - Set of pragmas for FSM and FSM state identification

SV-CC LRM Overview

- Coverage API – VPI Extensions
  - Obtaining coverage
    - Statement coverage from statement or instance handle
    - FSM iterators from instance
    - FSM coverage from instance, FSM handle, or FSM state handle
    - Toggle coverage from instance or signal handle
    - Assertion coverage from instance or assertion handle
  - Controlling coverage
    - Start, Stop, Reset, and Query on instance basis
    - Merge and Save per coverage type