



Simplifying the Behavior of SystemC Descriptions for Hardware/Software Covalidation

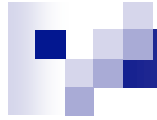
Hervé Alexanian, Sonics, Inc.

Gerald Bolden, Summit Design, Inc.



Problem Formulation/Statement

- The Coexistence of Different SystemC/
HDL Descriptions of the same Design
- Reuse of the Testbench Effort
- Correlate the Descriptions with Common
Stimulus



Agenda

- Definitions of Abstraction Level Terminology
- Designing the Transactional Testbench Interface
- Correlating the Behavior Descriptions
- Transaction Analysis and Debugging
 - Complete architecture analysis for various tradeoffs



Abstraction Level Terminology

- TL0: SystemC Representation of Signals
 - Can be mapped to HDL wires
 - `bool`, `sc_bv<N>`, `sc_lv<N>`
- TL1: Protocol Phase Accurate Model
 - Analogous to Bus Cycle Accurate
- TL2: Transaction Level
 - Analogous to Programmer's View with Timing Annotation



Designing the Transactional Testbench Interface

- Efficiently isolate the test program from the simulation environment
 - Different abstractions have incompatible interfaces
 - Choose a universal data structure to represent transactions
 - Decouple the test from DUV using transactors
- Testbench becomes reusable (untimed) component

Designing the Transactional Testbench Interface

- Transactors isolate test from DUV's abstraction layer

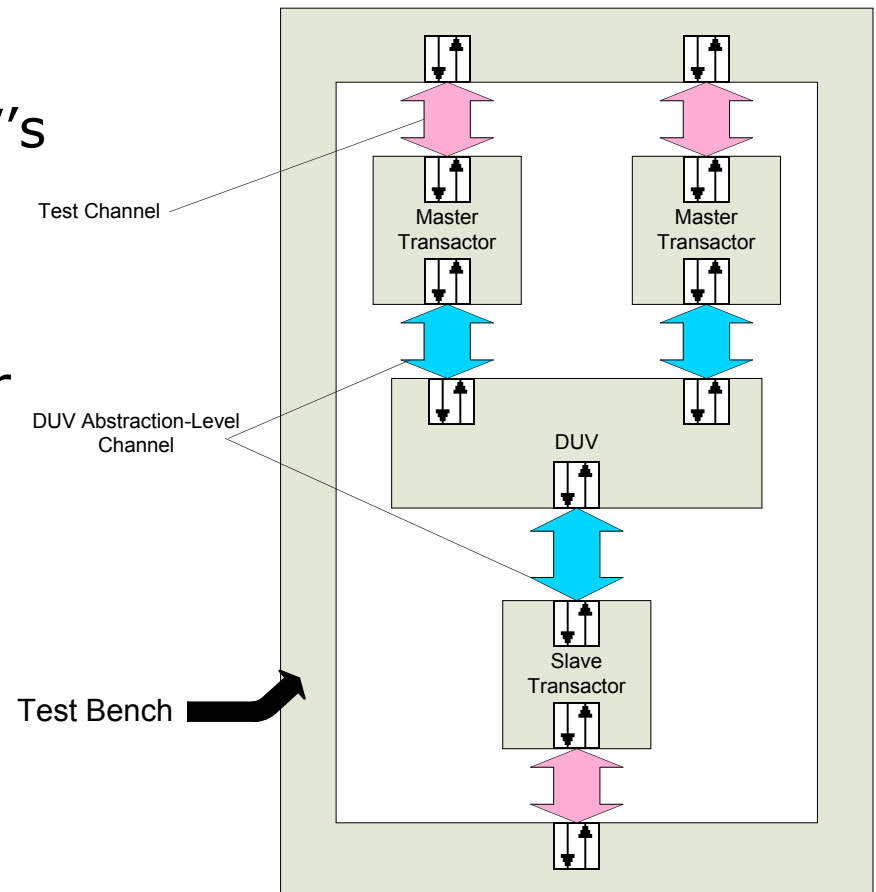
Provide standard test interface

Adapt to lower abstraction layer

- Test becomes a shell, which is:

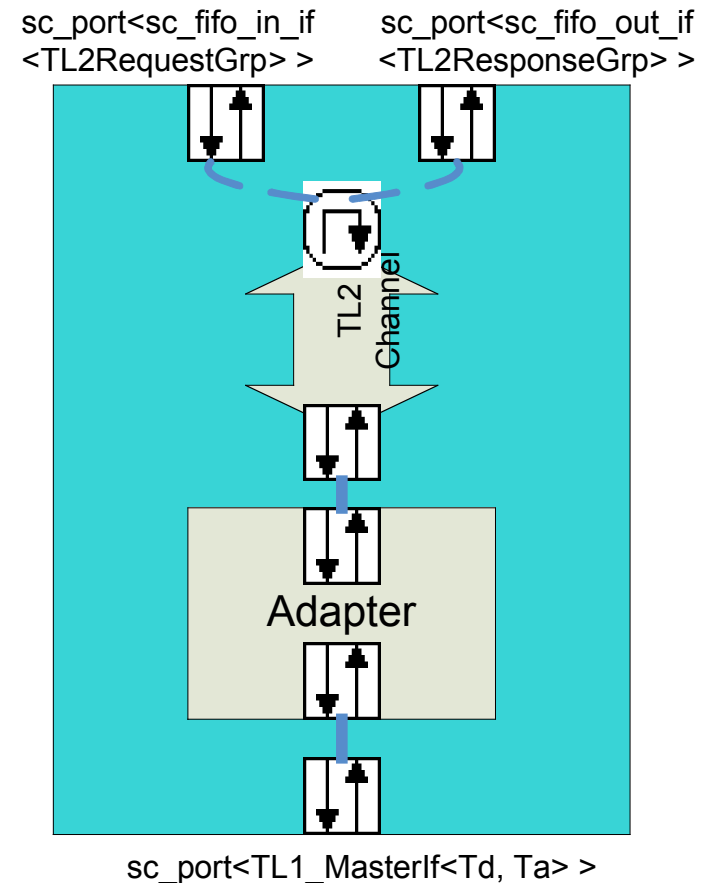
Only responsible for generating transactions

Reusable



Transactors

- Test interface uses only SystemC ports and methods
- Specific to DUV abstraction
 - Based on standard protocol components
 - Less effort than testbench



Simplified Transactor Behavior

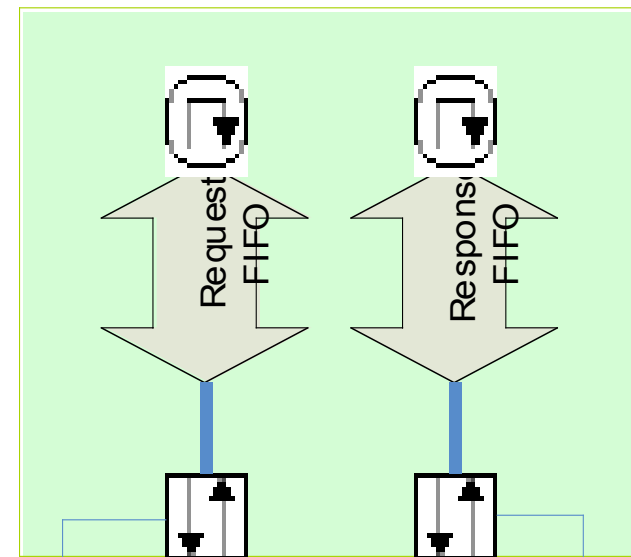
- Transactor reads from request FIFO port and sends as channel requests

```
void master_transactor_t12::request_thread () {
    bool result=false;
    while ( true ) {
        m_req = requestIn->read();
        cout << sc_time_stamp() << " : sending: " << m_req << endl;
        result=MasterP->sendOCPRequestBlocking(m_req);
        //check if the Request has been put into the channel
        assert(result);
    }
}
```

3

Transactional Test

- Generates requests/responses into FIFOs
 - SCV constrained random generation
- Good use of `sc_export` for FIFO interfaces
 - Simplifies the binding to transactors



`sc_export<sc_fifo_in_if<TL2RequestGrp > >`

`sc_export <sc_fifo_out_if<TL2RequestGrp > >`



Correlating SystemC Behavior Descriptions

- Validate Functionality

- SystemC TL2/SystemC TL1/RTL

- Validate Timing

- More difficult since models may have inherent inaccuracies



Correlating SystemC Behavior Descriptions

- End to End Checking
 - Correlates behavior
 - Write/readback
 - Data checking
- Transaction Monitors
 - Included within testbench
 - Can have trace dump for postprocess
 - Postprocess comparison can measure timing correlation



The Key Takeaways

- Innovative design of the testbench transactor interface simplifies the task of correlating the behavior descriptions of the same design
 - The transactor establishes and assures the correct integration between components of different abstraction levels
 - And, allows a progressively refined model to be easily integrated into a platform throughout the design cycle
 - Reuse of the test environment and code/scenarios for various design descriptions – results in actual verification time savings
 - An integrated platform will facilitate analysis and debugging w/ recording and analysis of transactions while still managing traditional signals in a waveform/transaction viewer.



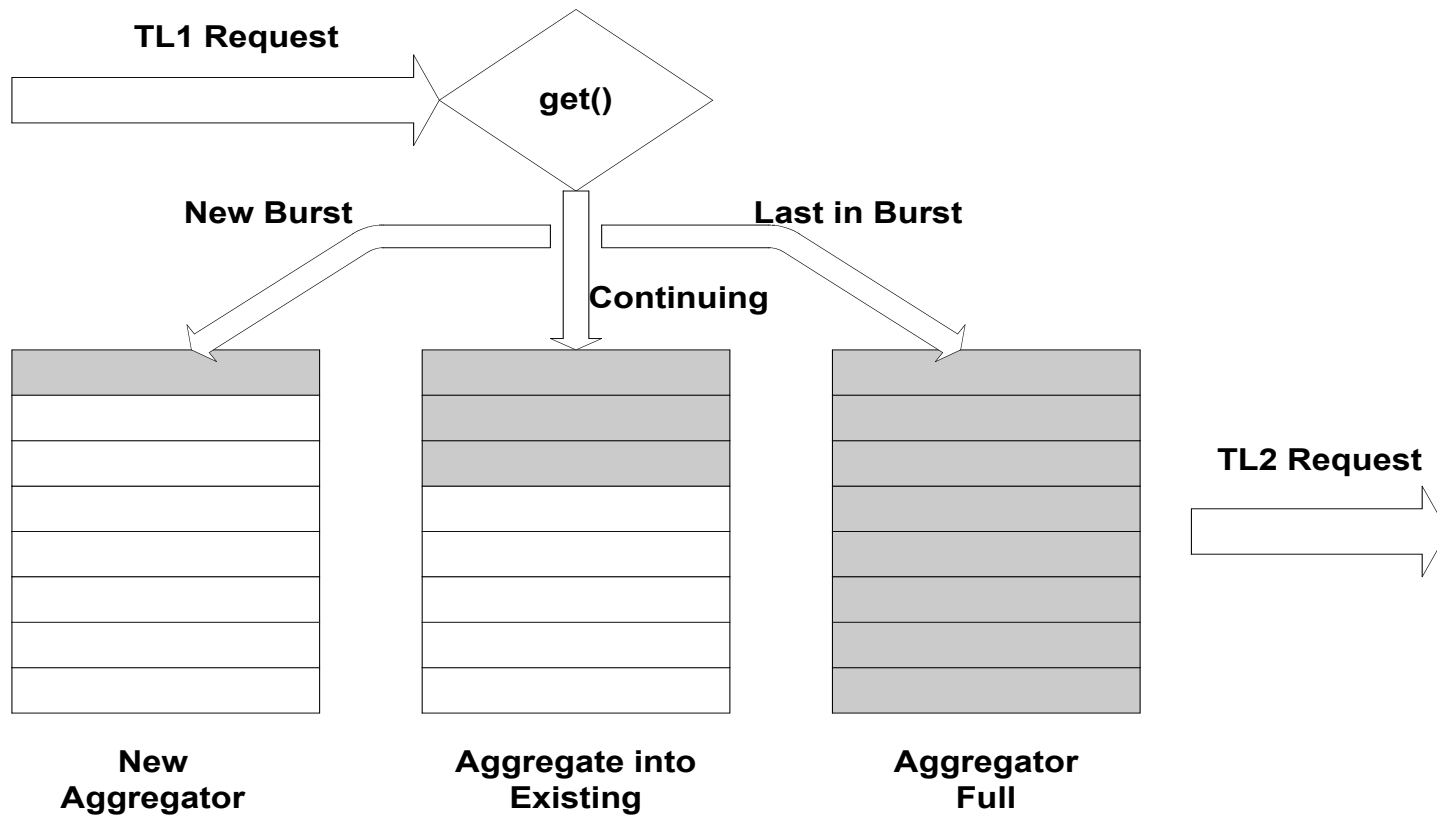
Thank You.

www.sonicsinc.com

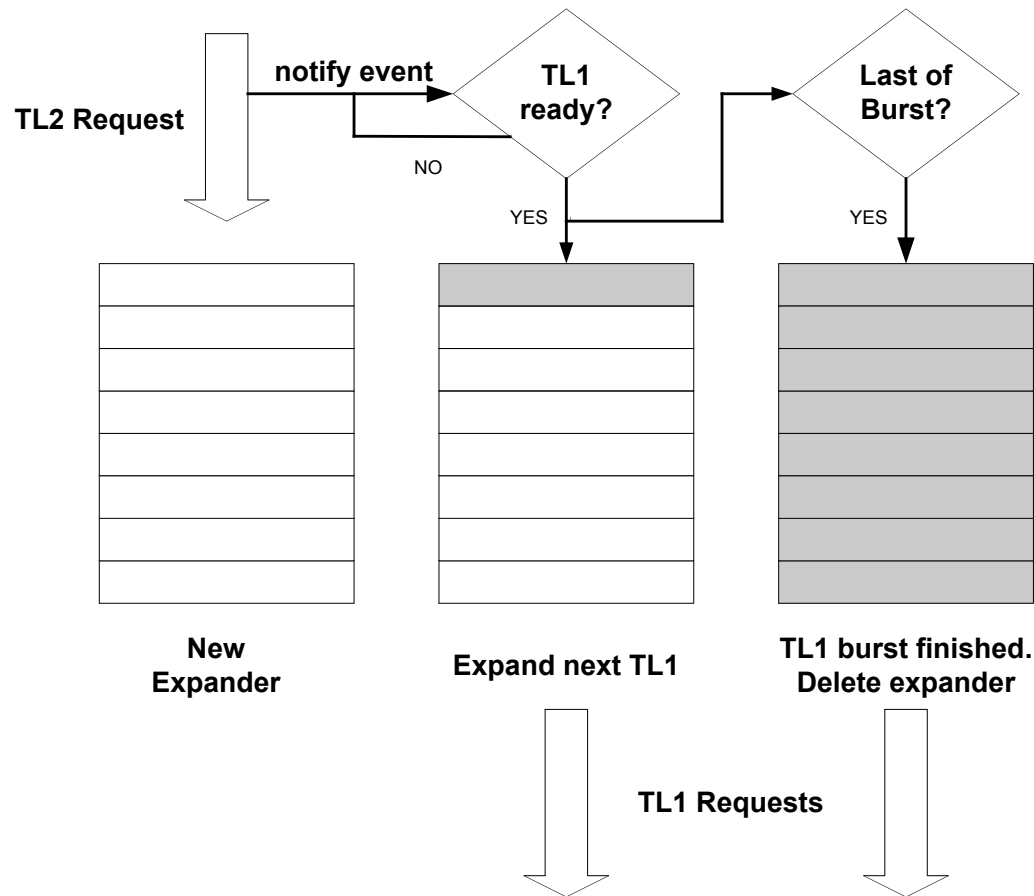
www.sd.com



TL1/TL2 Adapters



TL1/TL2 Adapters





What is Transactional Debugging, Anyhow?

- Defined as unified, parameterized, element which will represent a set of signals/data
- Transaction information fully compatible w/ the corresponding signal information in the same environment



Transactional Debugging

- Traditional waveform viewer, such that thousands of signals must be checked to ensure proper integration.
 - Each signal represents one line in the waveform viewer
- Trace function calls rather than signals; trace the calling sequence of these methods
 - When they're called and when they're returned, incl their argument values(e.g., read, add, data, and control) and timestamps (cf. delta cycles)



Transactional Debugging

- Manual analysis limited to defining the basic system architecture; and, using known parameters of contention
- Simulation tools required to perform complete traffic analysis and architecture tradeoffs regardless of timing implementation.