Bluespec and Co-Simulation

The First Bluespec Workshop

Monday, August 13, 2007

Myron King

Work done at NRC-Cambridge with:
Nirav Dave, Elliott Fleming, Gopal Raghavan, Jamey Hicks, and John Ankorn

Outline

- Description of the task at hand
- Some initial attempts (since discarded)
- How we are currently tackling the problem of Co-Simulation with Bluespec Compiler tools
- All in the context of the HS-TDEC (High Speed Turbo Decoder) implementation effort at NRCC this summer

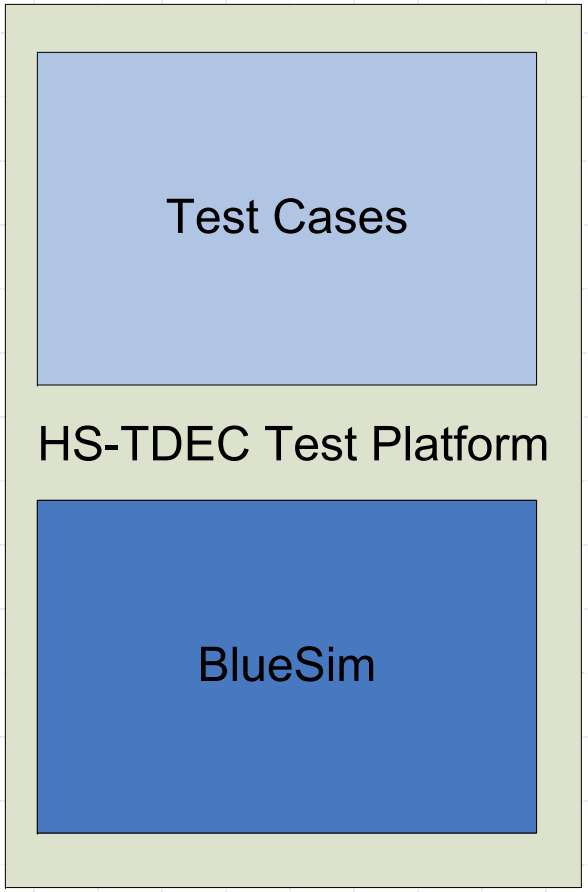
Nuts and Bolts

The Problem

- **Traditional HW Simulation is very slow**
- Reliance on massive Co-Simulation infrastructures
  - Complexity makes reimplementation prohibitive
  - Much effort put into speeding up RTL simulation
- What about C-level HW Simulation?
  - Carbon's RTL to C solution?
    - Not synthesizable
    - Equivalence of simulator and HW not guaranteed
  - **BSV and BlueSim**?
    - HW simulation C-speed (almost!)
    - Equivalence to generated Verilog guaranteed
    - Synthesizable

Replacing ModelSim with BlueSim

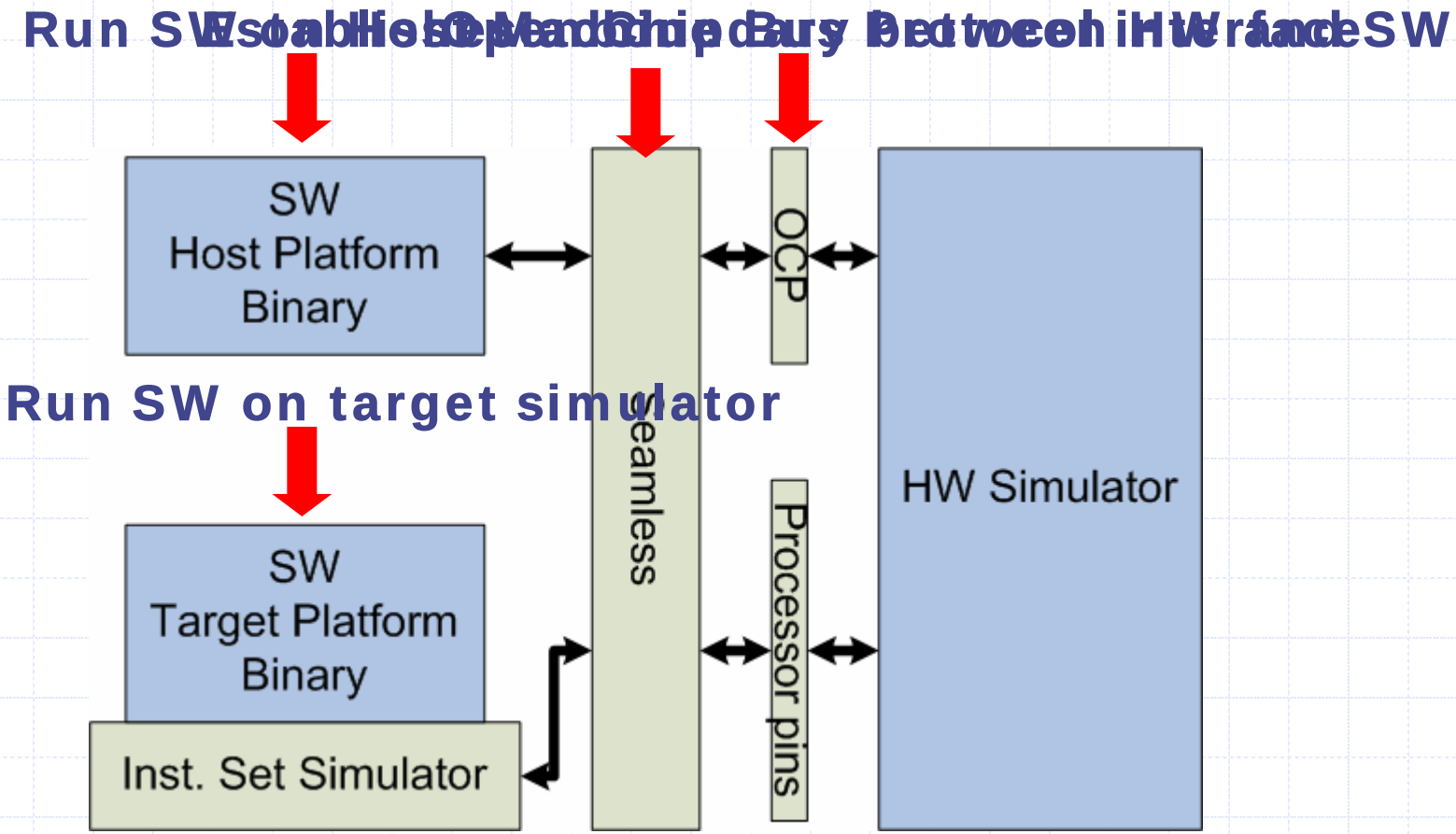
- We must fit Into Nokia's Test Framework
- Current HW written in VHDL
- Simulated using ModelSim in large testing environment.
  - **Unacceptably slow!**
- Replace ModelSim with BlueSim
  - **Our solution**



The Plan

- Connect BlueSim with Co-Verification Environment
- **Must support detailed simulation control**
  - **Module level API including clock & reset**
  - **Timed and functional verification**
- We have a Test bench
  - We need to use it
  - Our design must pass!

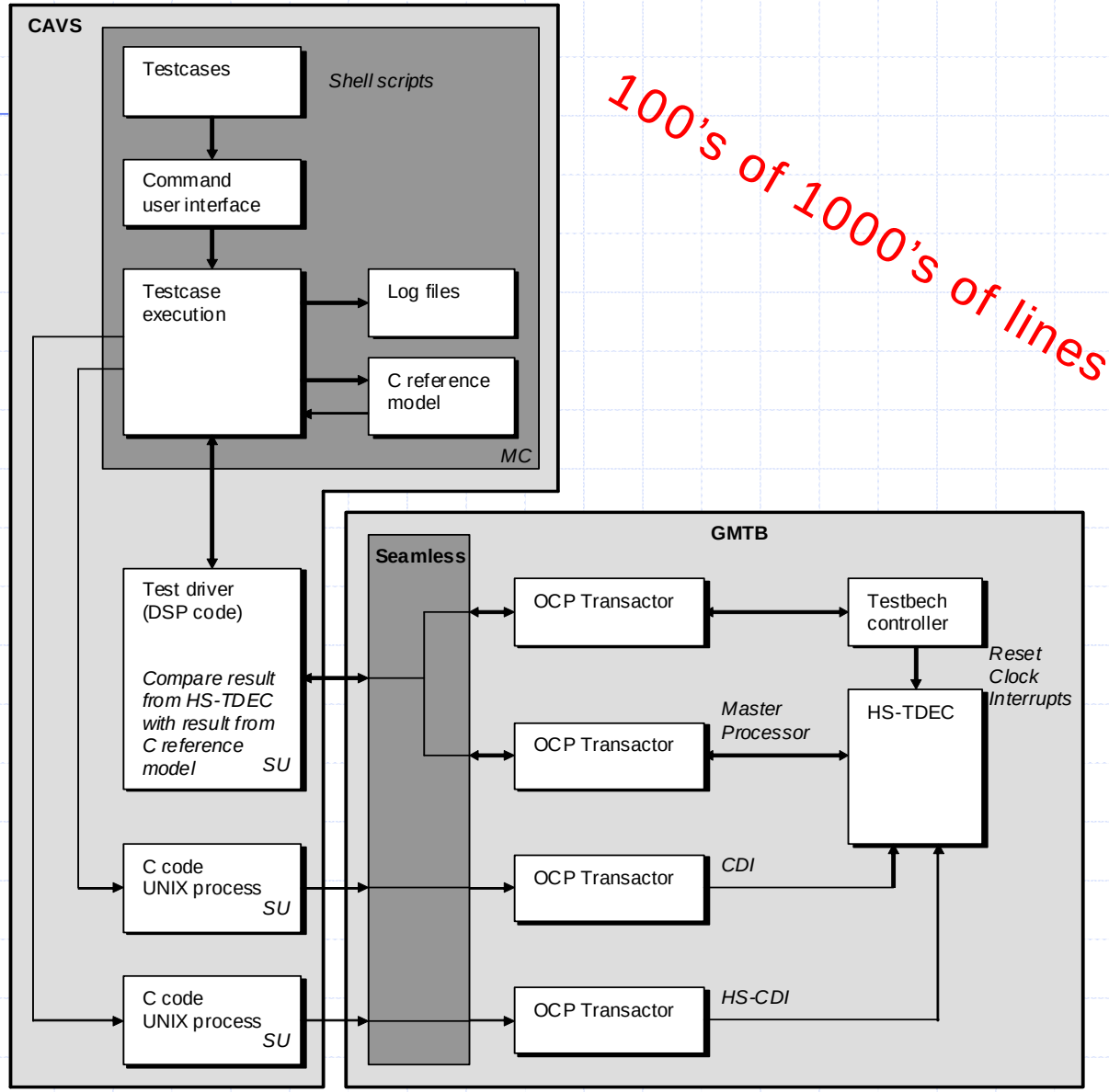
Seamless by Mentor



Some Issues

- BlueSim doesn't have a simulator interface
  - Should we build our own API?
  - How do we support module-level testing?
  - Easier for decoupled testing
    - Push data on a fifo, dequeue result when it appears
  - Need control over the clock for timed simulation
    - Requirement for timed simulations
- What is a simulator interface?
  - Clock control
  - Access to module interfaces
- All Bluesim gives you is an executable

Nokia's Test-Bench



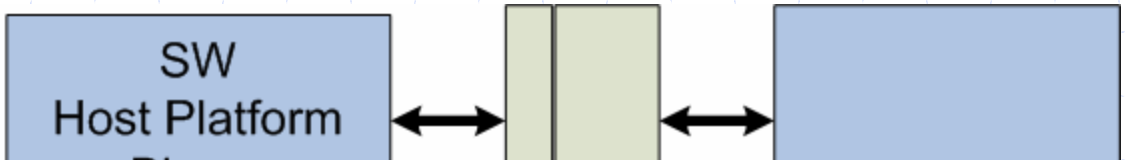
100's of 1000's of lines of code

First Attempt

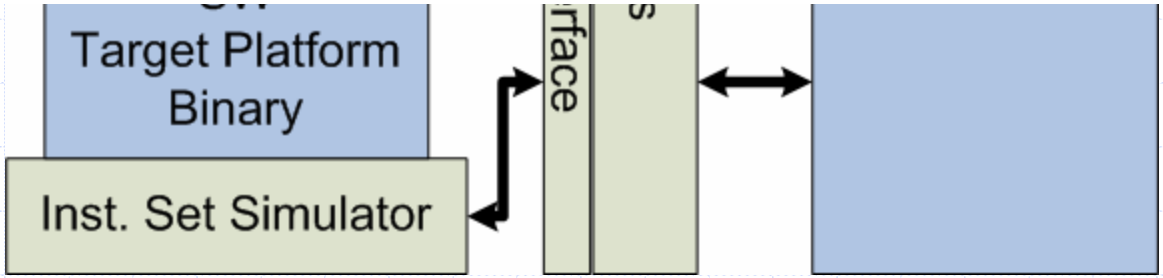
- New implementation Of Seamless SW interface
  - Implement required subset of functionality
- Guarded FIFOS implemented in C
  - All HW/SW interaction through these
  - Imported into Bluespec through BDPI interface
- Guaranteed not to break BSV semantics
  - But very conservative
- Doesn’t permit fine-grained modular interaction between Bluesim and C (restrictive)

Custom Seamless Implementation

Only FIFO-based communication



Abandoned when Gopal discovered the undocumented BSC –systemC flag!



Unable to support more complex OCP communication protocols

certainly no processor pin support!

Second Attempt

- Seamless supports the OSCI SystemC reference simulator
- Need to Link:
  - BSC-generated SystemC objects
  - OSCI SystemC kernel
  - Seamless systemC objects for comm. w/infrastructure
- Lots of complications getting everything to work together
  - Most resulting from new tool-flow combination

Bluespec Interface Restrictions

We'd like to do it like this:

```
interface CDI;
    method
    ActionValue#(Bit#1) cmdIn
    (Bit#(2) mAddr,
      Bit#(3) mCmd,
      Bit#(5) mData);
endinterface
```

We need to do it like this:

```
Interface OCPSlave;

  method Bit#(1) cmdAccept();

  method Action cmdIn(
    Bit#(2) mAddr,
    Bit#(3) mCmd,
    Bit#(5) mData);

endinterface
```

Bluespec Interface Restrictions

- No Combinational Loops through interface
- Only ValueMethods and Actions
  - Violations still possible
  - BSC will detect these
- Restrictions are due to scheduling issues
  - Bluespec team claims these are temporary
- Small Efficiency Hit
  - Single Cycle Bus Interface is now Multicycle
  - Sometimes requires extra registers + logic

Semantics of SystemC

- TLM is being promoted as modeling methodology for SystemC
  - Semantics not well Understood
- TRS is the semantics we want
  - Well understood
  - We think this gives the proper semantics of TLM
- Current picture some sort of unholy union
  - Not sure what **exactly** is happening here

Question

- Do we want to TRS in SystemC?
  - Overhead of rule scheduling logic is large!
  - Sequential Connective
    - Real SW generated from rules (Nirav's thesis!)

In Conclusion

- BlueSim SystemC integration is proving Very useful
- It's convenient to write test-benches in C and drive the HW model through exposed interface
- You don't sacrifice simulation speed and still get all the goodies from SystemC
- We have this up and running at Nokia