Simulation Acceleration through Hardware Emulation
Transaction Based Verification Environment for 32 bit Booth's Multiplier
(Co-Simulation with Co-Modeling using TLM SCEMI Pipe Interfaces)

by
Sameer Ghewari
March 20, 2014

Introduction:

This project demonstrates development and execution of Co-Simulation with Co-Modeling mode of verification environment for Mentor Graphics Veloce emulator. The DUT is a 32 bit Booth's Multiplier.

The DUT and transactor are compiled targeted to Veloce Solo. The transactor accepts operands from HVL side running on the workstation in Questa through SCEMI 2.0 TLM pipe interface. The SECMI 2.0 pipes are FIFOs and are provided as xtlm package in Veloce. The HVL testbench is written in SystemVerilog and pushes Constrained Random operands onto the pipe.

The transactor picks up these operands and excites the DUT with these. The DUT takes 32 clock cycles to produce one multiplication result. The result is sent back to the HVL through output instance of SCEMI pipe. The process repeats for the number of pushed operands onto the input pipe by HVL. The HVL produces output log files containing multiplicand, multiplier, and result. It also indicates if the obtained result matches to the expected one.

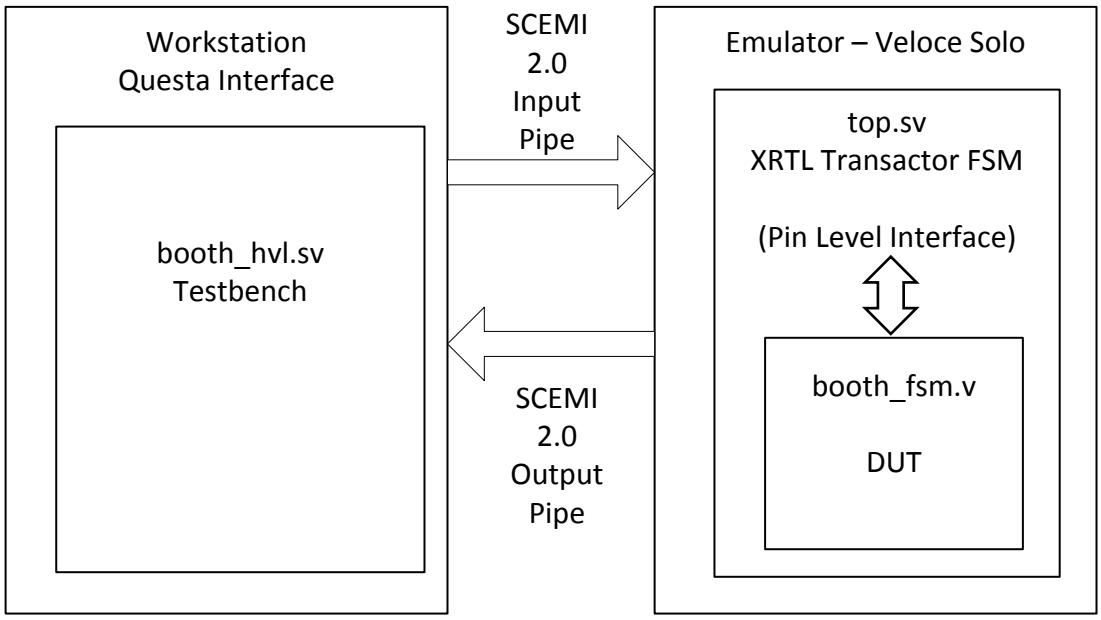

Fig 1. Testbench Architecture

The DUT- Booth's Multiplier

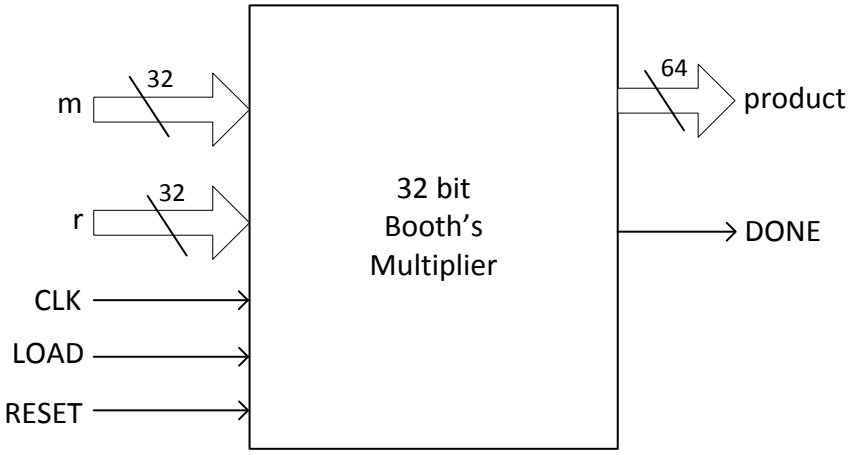

Fig 2. Block Diagram of Booth's Multiplier

A 32 bit Booth's Multiplier has been modeled in Verilog as an FSM. The following graph illustrates operations of this FSM. No performance optimizations are made for the simplicity of the DUT.
The module samples two 32 bit numbers – Multiplicand (m) and Multiplier (r) when the LOAD signal is high at a posedge. The result of the multiplication is available after 32 clocks. Once result is ready, it is put on 64 bit Product bus and DONE signal is pulsed.

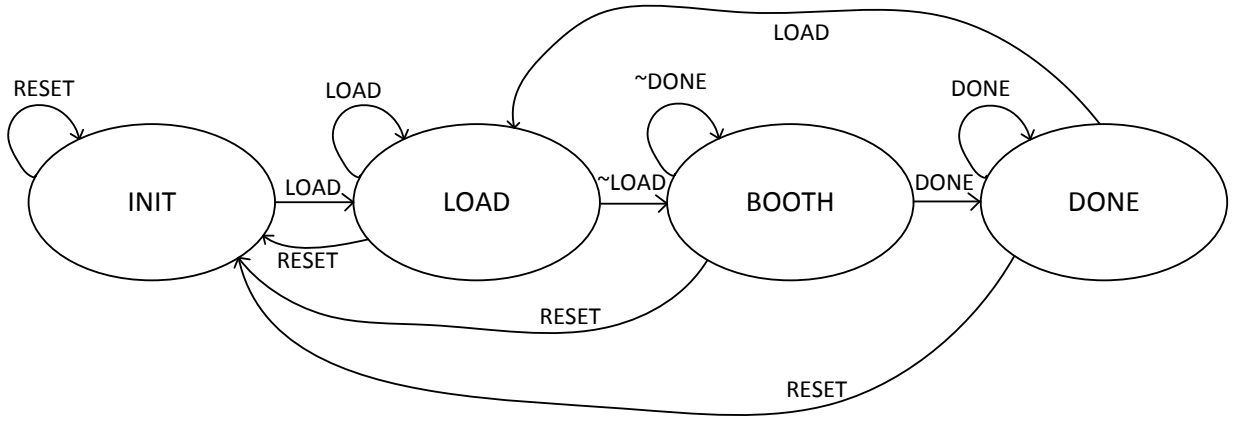

Fig 3. State Transition Graph of Booth's Multiplier Algorithm

Setup:

1. Connect to velocesolo.ece.pdx.edu on port 22 with X forwarding enabled. Make sure that xming or similar client is running in the background.

2. Copy the entire directory with files to the Veloce workstation. The file structure should look like the following-

project_folder/tbx.config
This file contains important configuration settings about design partition and target emulator

project_folder/config.v
This file contains parameter for multiplier width

project_folder/Makefile
This is the makefile to compile, build and launch emulation

project_folder/hdl/booth_fsm.v
This is the FSM for Booth's Multiplier. This is the DUT in this example.

project_folder/hdl/top.sv
This is the top level DUT and contains XRTL FSM transactor. Also contains instantiations of SCEMI TLM pipes and XRTL clock generation logic.

project_folder/hvl/booth_hvl.sv
This is the testbench that would run on the workstation. It contains SystemVerilog code to generate Constrained Random operands that are pushed onto the pipe. It also monitors output.

Simulation (Puresim Mode)

It is advisable to run the environment in puresim mode first. In the puresim mode, entire environment runs on Questa workbench as a complete simulation. However the SCEMI transaction pipes are used to transfer data between the HVL and HDL.

The puresim mode uses TbxSvManager to interconnect HVL and HDL through pipes.

1. To compile, build for puresim mode-
   >make puresim

make puresim invokes-

```
vlib puresim_work                          //Create new library for puresim mode
vmap work puresim_work                     //Mount the new library with default
vlog –f $(TBX_HOME)/questa/hdl/xtlm_files/f  //Compile xtlm files, (SCEMI pipe modules)
vlog hvl/booth_hvl.sv                      //Compile HVL testbench
vlog hdl/top.sv hdl/booth_fsm.v            //Compile HDL transactor and DUT
tbxsvlink –puresim                         //Setup TbxSvManager for puresim mode
```

2. To launch the simulation
   vsim –novopt top booth_hvl TbxSvManager +RUNS=<NumofTests> +SIGNS=<SignsofOperands>

Where Num of Test = The number of test cases you would like to generate <integer>
and Signs of Operands = The sign of multiplicand and multiplier you desire <++, --, +-, -+>

e.g: vsim –novopt top booth_hvl TbxSvManager +RUNS=100 +SIGNS=++
Above command launches simulation and generates 100 test cases where Multiplicand and Multiplier both are positive.

This launches a Questa interface. The signals may be added by – add wave *
Enter – run-all to simulate for all operands

3. If the simulation in puresim mode was successful, in the project_folder the following three files are generated.

   Multiplicand.txt – The list of generated multiplicands
   Multiplier.txt – The list of generated multipliers
   Product.txt – The list of obtained product

Emulation (Veloce Mode)

1. In the Emulation Mode, TbxSvManager connects the HVL running on workstation and HDL running on the Veloce through pipes.

   To compile, build and simulate in puresim mode-
   >make veloce

make veloce invokes-

```
vlib veloce_work                                  //Create new library for puresim mode
vmap work veloce_work                             //Mount the new library with default
vlog –f $(TBX_HOME)/questa/hdl/xtlm_files/f       //Compile xtlm files, (SCEMI pipe modules)
vlog hvl/booth_hvl.sv                             //Compile HVL testbench
tbxcomp –top top hdl/top.sv hdl/booth_fsm.v -veloce  //Compile the DUT for Veloce target
tbxsvlink –puresim                                //Setup TbxSvManager for veloce mode
```

2. Before launching the emulation, make sure no other user has occupied the emulator by-
   >whoison –emul velocesolo1

3. To launch the simulation
   vsim –novopt top booth_hvl TbxSvManager +RUNS=<NumofTests> +SIGNS=<SignsofOperands>

Where Num of Test = The number of test cases you would like to generate <integer>
and Signs of Operands = The sign of multiplicand and multiplier you desire <++, --, +-, -+>

e.g: vsim –novopt top booth_hvl TbxSvManager +RUNS=100 +SIGNS=++
Above command launches simulation and generates 100 test cases where Multiplicand and Multiplier both are positive.

This launches a Questa interface. It takes a while for the design to be downloaded to the emulator.
Enter – run-all to simulate for all operands

4. If the simulation in veloce mode was successful, in the project_folder the following three files are generated.

   Multiplicand.txt – The list of generated multiplicands
   Multiplier.txt – The list of generated multipliers
   Product.txt – The list of obtained product

References

1. Faust, M., “Using Mentor Veloce at PSU”, Aug 2010
2. Chung, H., “Veloce Stand-Alone Mode Tutorial”, Sep 2010
3. Mentor Graphics, “TBX-Veloce Training Slides”
4. Mentor Graphics, “TestBench-Xpress User Guide – Software Version 2.3.1.3”, Nov 2009
5. All other documentation on Veloce Solo, as required