CASPER Memo 016
Co-Simulation Setup Procedure

Ben Blackman

August 2007

1 Motivation

This is a guide to co-simulating verilog versions of simulink blocks. The goal of this procedure is to verify that the verilog is both bit-accurate and cycle-accurate with the simulink design. This procedure was used to test the Mars Spectrometer, which was an ASIC version of the FPGA-based Pocket Spectrometer.

2 Capturing Test Data

1. Go into the simulink design and locate the block you wish to test.
2. Connect *To Workspace* blocks to all inputs and outputs of the block. Double click the blocks to give the data appropriate names. Note: if an input or output is really 2 pieces of data concatenated (Ex. real and imaginary parts of a number), it may be easier to slice the data and record each separately.
3. If the test data is a recognizable pattern (Ex. a ramp or sine wave), then you may want to also place scopes on the inputs and outputs.
4. Simulate the simulink design for an appropriate number of cycles.
5. Go to MATLAB and click on the Workspace tab in the Current Directory window. Select the desired pieces of data and save them as a .mat file.

3 Setting up the Co-Simulation Model

1. For each block to be tested, there is usually a .v and .sdf file (Ex. mars_008_tvg.v and mars_008_tvg.sdf). The .v file contains the module and the .sdf contains timing specific information about the module. Make a copy of the .v file and rename it, such as *mars_008_tvg_template.v*. Open this file, and delete everything except the mars_008_tvg module.
2. Open a new model file and create a "black box" block. Simulink should ask you for a verilog file. Select the template file. The black box should now have the same inputs and outputs as the block you are simulating, and there should be a config.m file that opens up.
3. If the inputs or outputs are not in the same order as the simulink block, you can change the order in the config file. Also, towards the bottom of the config file, there will be a line similar

to, *this_block.addFile('mars_008_tvg_template.v');*. Comment this line out using % because you will use scripts to run the simulation.

4. Now add the System Generator and ModelSim blocks to the model. Double click the ModelSim block and set the co-sim directory to *./modelsim*. Check Open Waveform Viewer, Leave ModelSim Open at End of Simulation, and Add Custom Scripts. Do not check Skip Compilation. For the following fields, give the following tcl files:
Script to Run Before Compilation - *build_modelsim_libs.tcl*
Script to Run in Place of "vsim" - *do_vsim.tcl*
Script to Run After "vsim" - *post_vsim.tcl*
For details about these scripts, see *The Tcl Scripts* section below.

5. Because we want to test the behavior of the black box using ModelSim, you will need to change the sample period. The easiest way to do this is to go to Model Properties->Callbacks tab->InitFcn. Here type *sample_period = 1e-6;*, which sets the sample period to be 1 microsecond. If you want a different period then do so but the design may not function properly if the sample period is too small. This will create a Workspace variable called sample_period for your design to reference.

6. Now you want to set the sample period of every block on this clock to be sample_period. If simulink complains that sample_period doesn't exist, then it is because sample_period won't be created until you simulate your design. To fix this, you need to just go to Workspace and create a new variable, sample_period, and set its value to *1e-6* or the value you're using. Be sure to open System Generator and set Simulink System Period to sample_period.

7. Add the *From Workspace* blocks and match their names with the input data you saved with the *To Workspace* blocks from data capture step. Add *To Workspace* blocks for the outputs of the black box.

8. Double click the black box and set:
Simulation Mode - *Use HDL Co-Simulation*
HDL Co-Simulator To Use - *ModelSim*

9. Now set the simulation time to be the number of clock cycles set from before multiplied by sample_period. Also, find the .mat file with the input and output data and load that into the workspace.

10. There should be a *rst* port in the asic version which you should give a step starting with 1 and ending at 0. It should go to 0 before the input sync.

11. Simulate and compare the output data in MATLAB.

12. It is possible to simulate the asic verilog version along side the simulink version. You have to wire the inputs the same and change the sample period fields in the simulink version to sample_period because they are probably set to 1 by default (it can be very tedious to change every block and might not be worth it).

4 The Tcl Scripts

4.1 build_modelsim_libs.tcl

Make sure the references to paths are correct, like PROJECT and others. Also, the first 4 vlog commands build libraries and must be compiled only once. The last vlog command builds the black box verilog and must be compiled the first time and then only when the verilog is changed. When they don't need to be compiled, comment out the vlog lines with #.

4.2 do_vsim.tcl

Make sure that PROJECT and TOP_DESIGN paths are correct and name the black box *dut* in your design. This stands for device under test and is done because dut_path is set up to find the black box named *dut*.

4.3 post_vsim.tcl

Make sure the TOP_DESIGN path is correct. The rest of this file is up to you. This allows you to specify what waves show up in the ModelSim wave viewer. If you don't know what waves to put, run the simulation for a short time and then look in ModelSim for objects that you can add to the wave viewer.