DESIGN VERIFICATION Exercise 3:
How to collect Code Coverage with ModelSim

This exercise introduces you to the Code Coverage viewer of ModelSim. It is assumed that you have done the "Introduction to the ModelSim Simulator" exercise and hence know how to run ModelSim without collecting code coverage.

If there are problems, please let me know. Have fun!

Kerstin

Collecting Code Coverage with ModelSim

1. Set up your working library as usual. Before you compile your design, e.g. the `mux_int_test.v` testbench from the previous exercise, you must enable the respective source code coverage options (e.g. Statement Coverage, Branch Coverage, etc) you want to collect during simulation from the "Coverage" tab of the "Compile Options" menu . Coverage collection must also be enableded at the time when you load a design unit. Then complie your design as usual.

   From the "Simulate" menu select "Start Simulation". In the "Start Simulation" window select the "Others" tab and tick "Enable code coverage". (Notice that you can also specify the name of the dataset file for storing waveforms in this window.) Then select the "Design" tab, select the design to be loaded and disable optimizations. Several coverage statistics windows appear.

   Watch the console window for the command-line instructions.

2. Run the simulation and investigate the code coverage results. (You almost certainly will have to enlarge the "sim" pane "(Local Coverage Aggregation)" to see all results.) Check the Statement Count, the Statement Hits, Statement %, Statement Graph, etc.

   The ModelSim SE User Manual (available in HTML format from the "Help" menu under "SE Documentation") contains further information on collecting code coverage and explains all the coverage statistics windows, how to save and merge coverage results, and other interesting features of the coverage collection tool.

3. You can browse the design hierarchy and see the total code coverage of each sub-unit as well as a detailed analysis of which lines have been covered.

How well have you exercised the calc1 design?

Part of the 2nd assignment is to find out how well the test suites you submitted for the first assignment, for which the calc1 source code was not available, exercised the calc1 design.

The source code of calc1 will be available in time for the 2nd assignment. It is wise to run your tests again and to measure code coverage. (You might just find a part of the calc1 design which you have not exercised, and more bugs could possibly be found there.)