

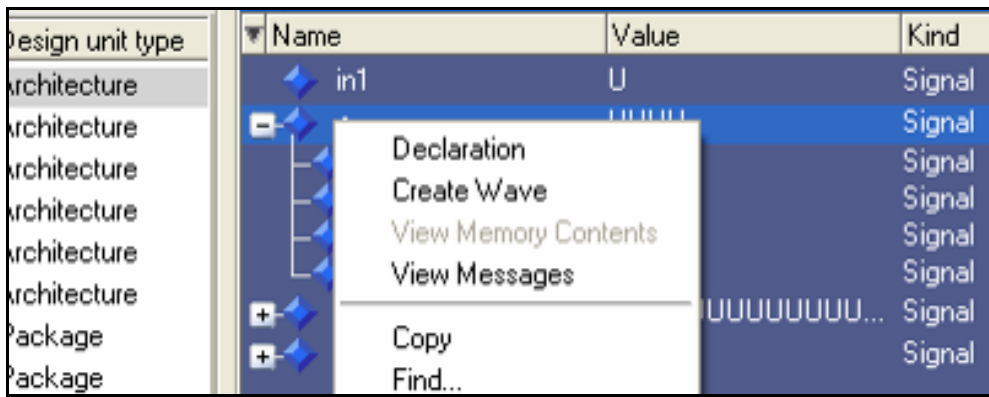
How to streamline the simulation process

Greg Galloway
Spring 2007

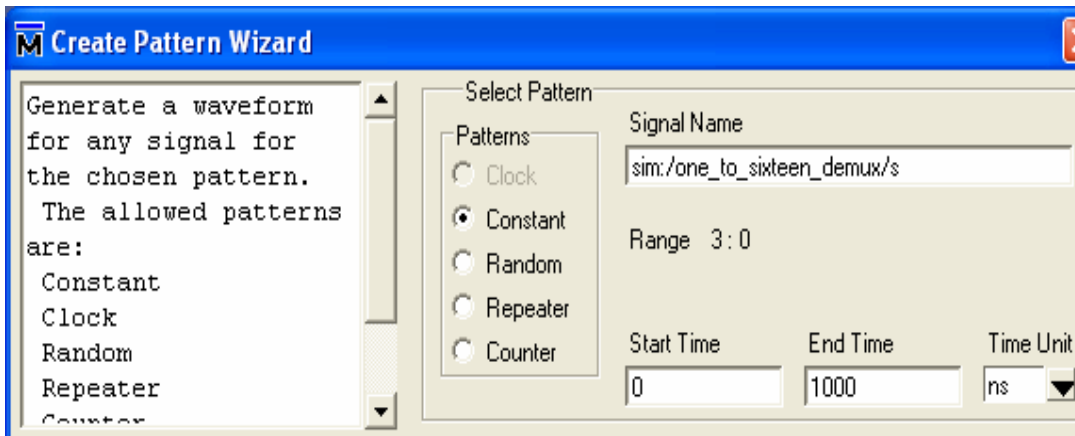
For this demonstration, I am using a one-to-sixteen demultiplexer, but this process applies to any unit under test.

Before setting up waves, make sure your entity is in a workspace you have access to. Then simulate it. The signals should appear in the middle window. This is the point where I will begin the explanation.

- 1.) First I **Right click** on the signal or set of signals that I wish to automate.
- 2.) Then I select **Create Wave**.



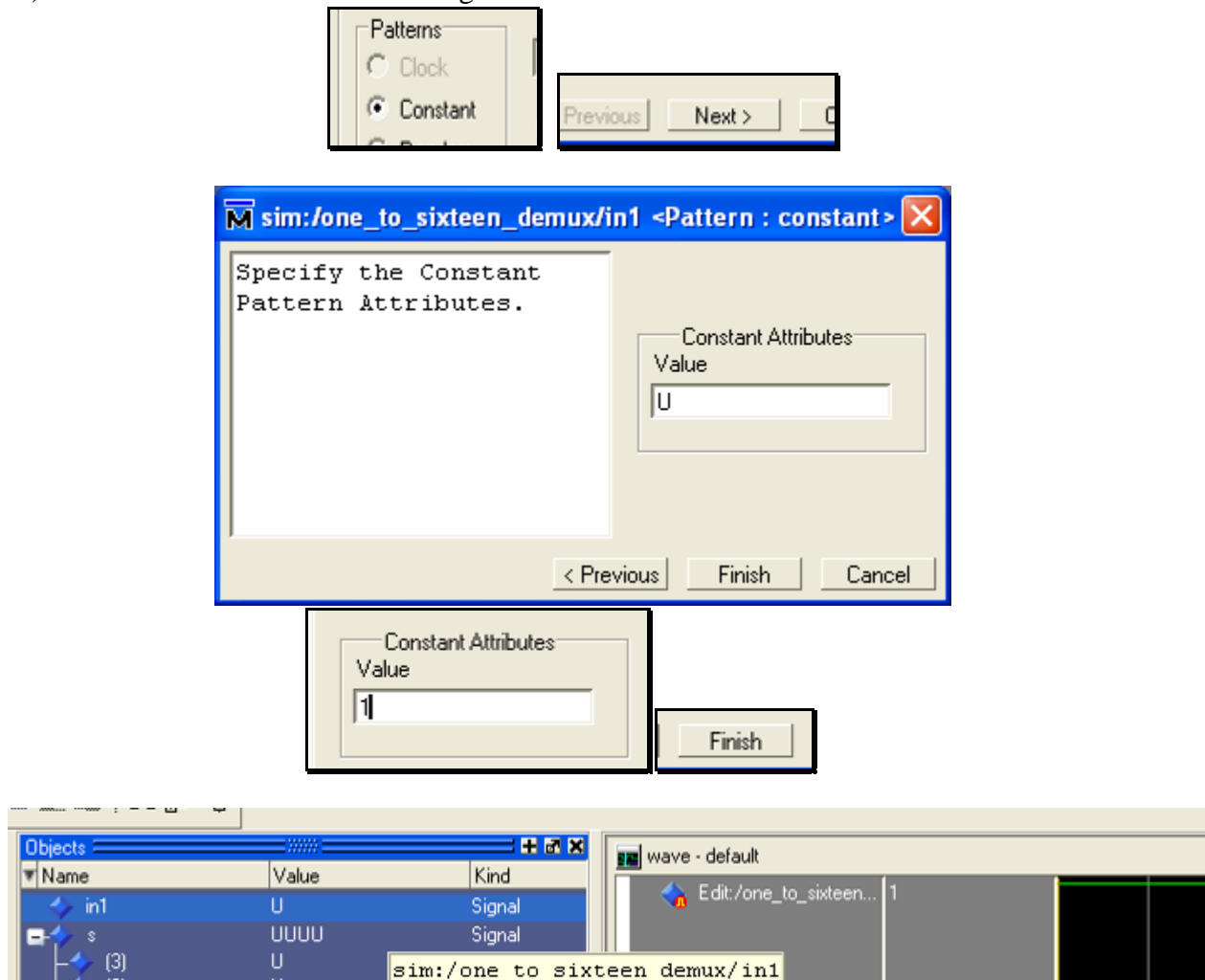
At this point I see the *Create Pattern Wizard*:



Briefly:

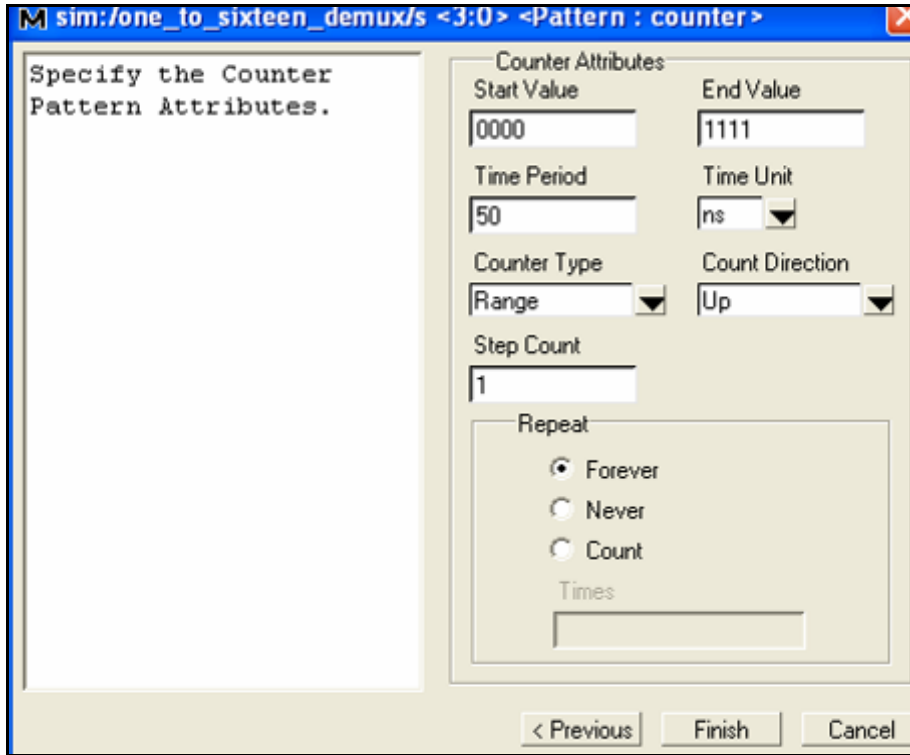
- **Constant:** Sets the signal to constantly put out one value, for this example a constant 1 on the input bit helps demonstrate that the input is reaching the right output from the demultiplexing. I will use this for **in1**.
- **Clock:** Not generally allowable, but not generally needed, either. If a constantly alternating signal is desired, a repeater or counter can be used.
- **Random:** If you need it, you'll know it.
- **Repeater:** Useful for generating cyclical signals, or for when a certain pattern is needed, as the name insinuates. For example, 0010 can be repeated as 0010001000100010...
- **Counter:** Good for iterating through all values that are possible, especially if the set of test values is limited. This is what I will use for the select bits in this example.

3.) I will first set **in1** as a constant signal of one.



Note that it lists the value as **U**, this is at time 0 and will not be a problem during the simulation. Also look to the right and see it is set as a constant value of 1 for all time.

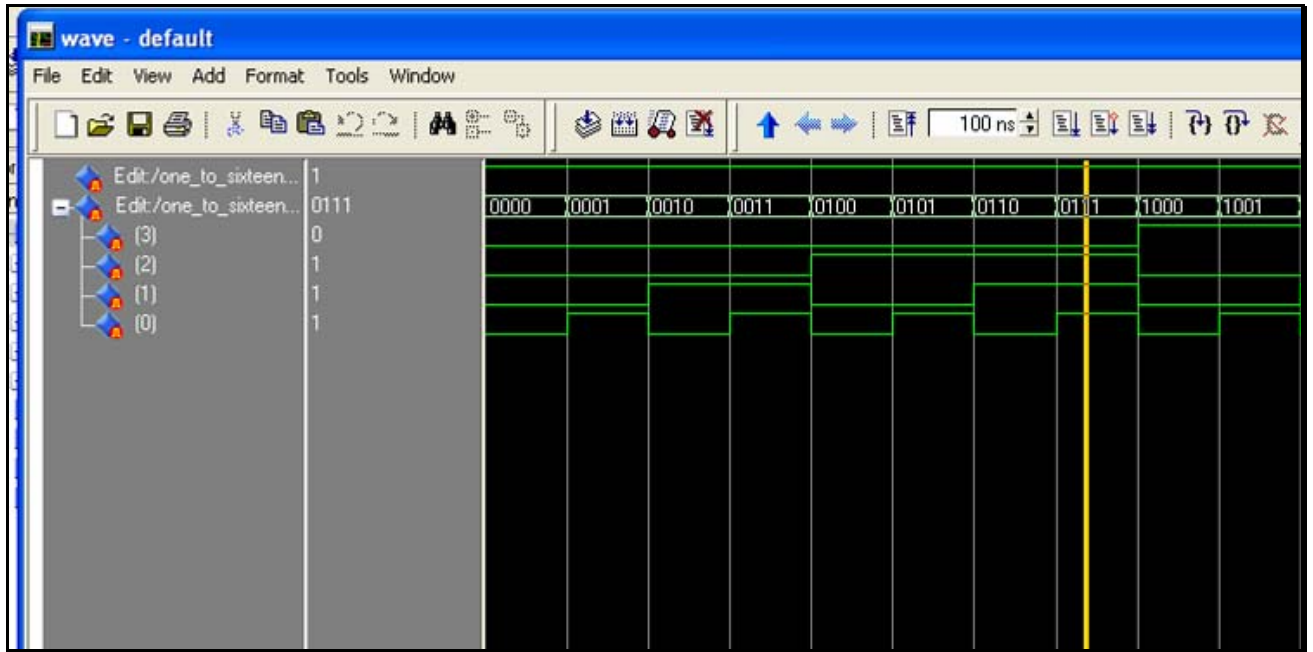
4.) Now I **right click** on 's' which is a four bit signal. I select **create wave and counter**.



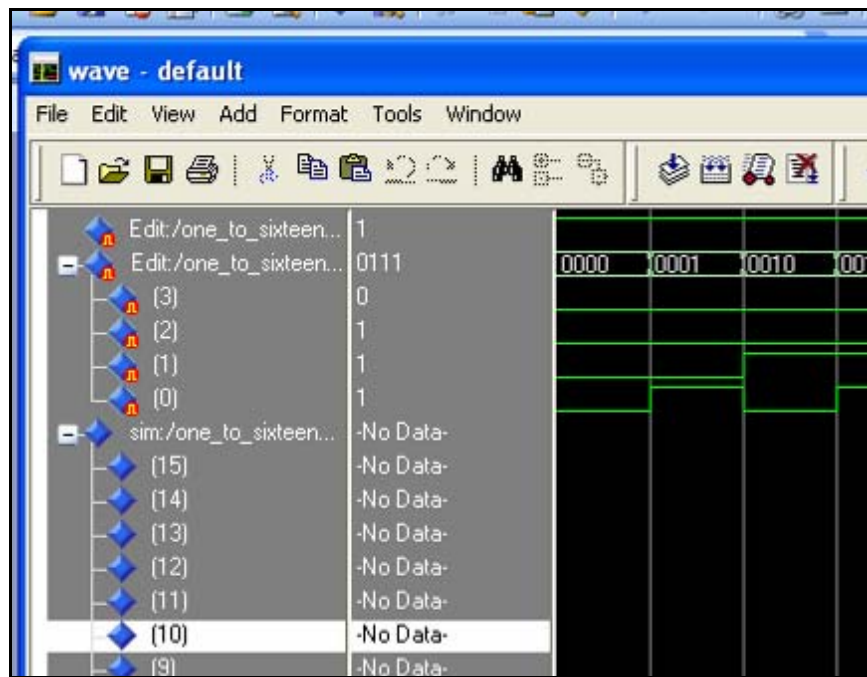
The defaults are fine for this example, and it will increment from 0000 to 1111 by 1 step every 50 nanoseconds. It will additionally repeat forever. As you can see, there is room for a lot of customization. But I will click finish at this point.



Now the *wave* window displays the following:

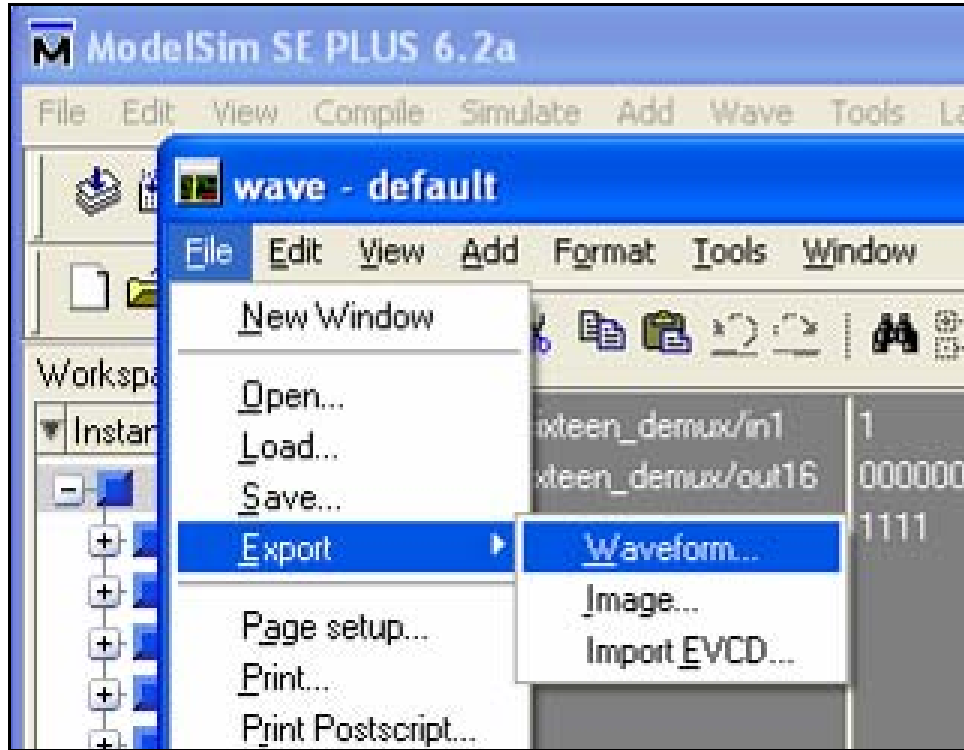


- 5) Now that my inputs are determined, I drag signal 'out16' to my *wave* window so I can have something to view when I simulate.

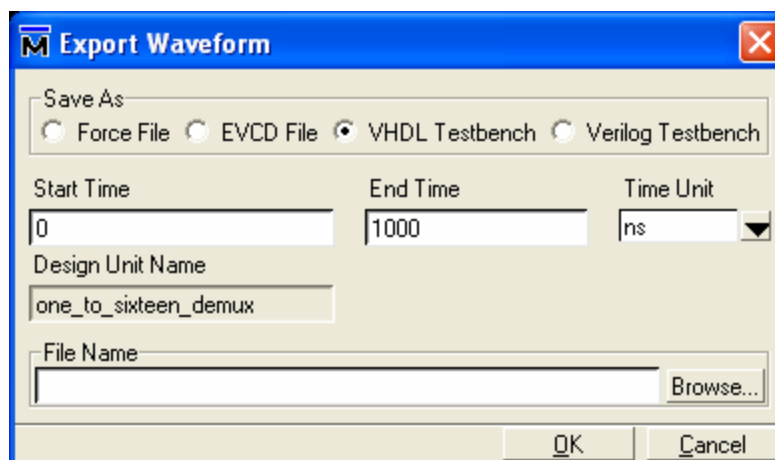


Storing the data:

Creating a test bench waveform:



File, Export, Waveform in the *wave* window.

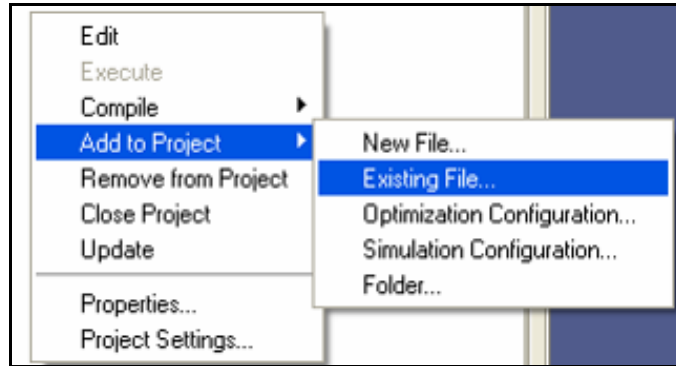


Select **VHDL Testbench** in the *Export Waveform* window.

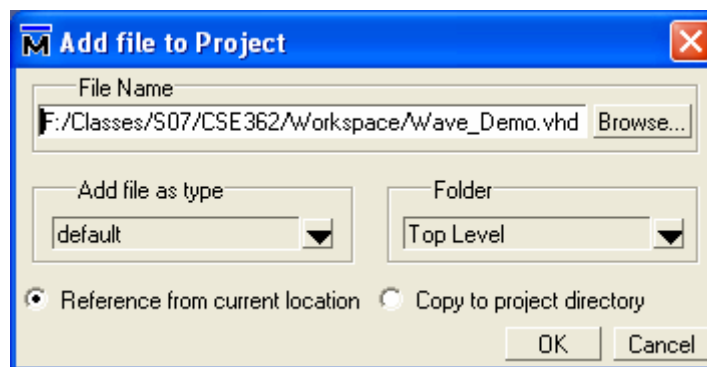
Browse to the directory you are going to save to, and select a file name. It should have the extension **.vhd**.

Restoring a waveform:

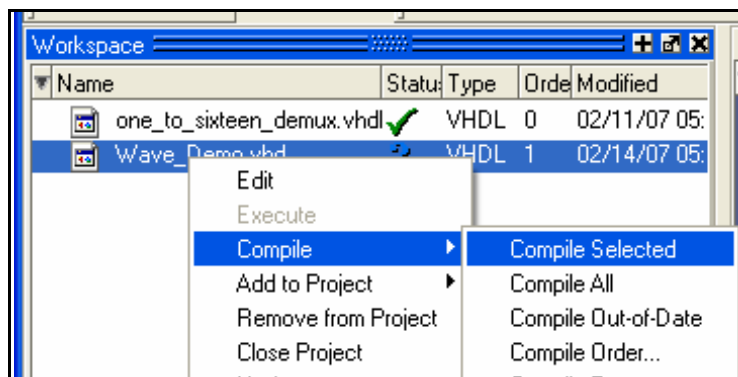
In the *project* tab of the *workspace window* right click, select **add to project** and **existing file**.



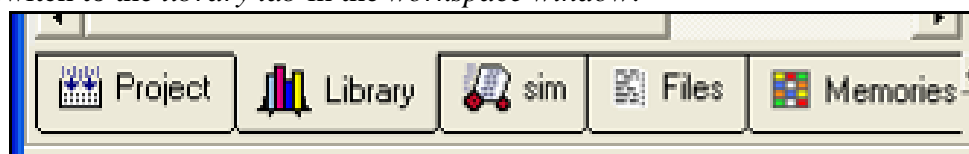
Select the file of the waveform you want to view.



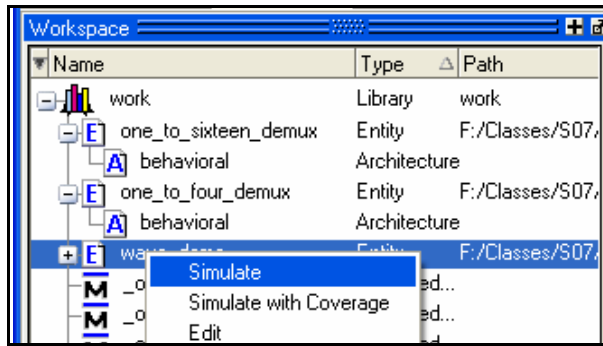
Compile it.



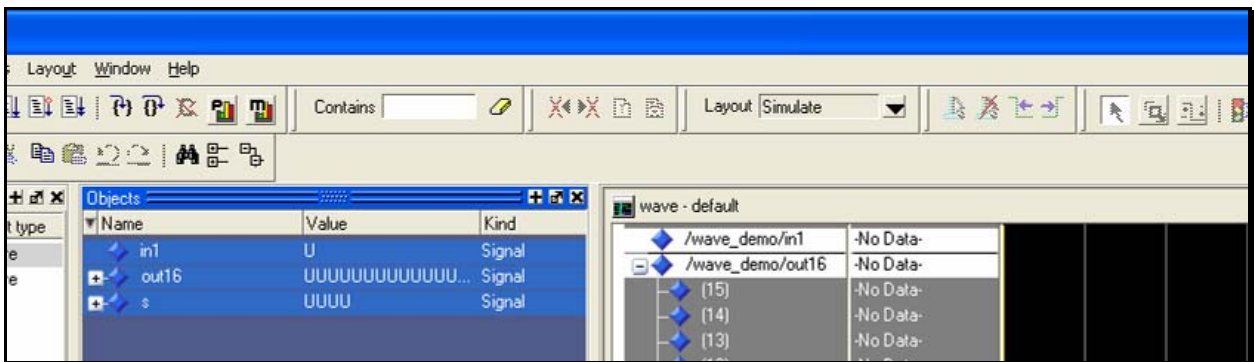
Now switch to the *library* tab in the *workspace window*.



Simulate the vhdl.



Drag the signals over to the *wave window*.



Run all and view the recovered information.

