ENGR 5865 DIGITAL SYSTEMS

ModelSim Tutorial Manual

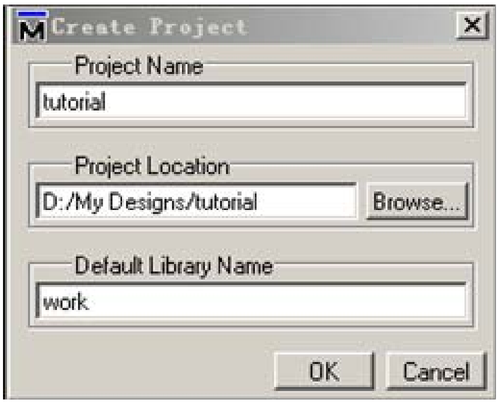
January 22, 2007

Introduction

ModelSim is a CAD tool widely used in the industry for hardware design. This document describes how to edit/add, compile and simulate VHDL code by using ModelSim software. The example code for this tutorial is a simple counter without testbench but setting the values of selected input signals manually in the waveform. Normally, a complete project should include a testbench for each module to instantiate the entity under test, and to initialize the input signals to the module. You may feel like creating your own testbench and simulate it after you complete this tutorial.

Create a Project

In the file menu, choose File -> New -> Project
This will bring up the following window:



Create Project

Project Name: tutorial

Project Location: D:/My Designs/tutorial

Default Library Name: work

Put the project in your preferred directory. Leave *work* as the default library name.

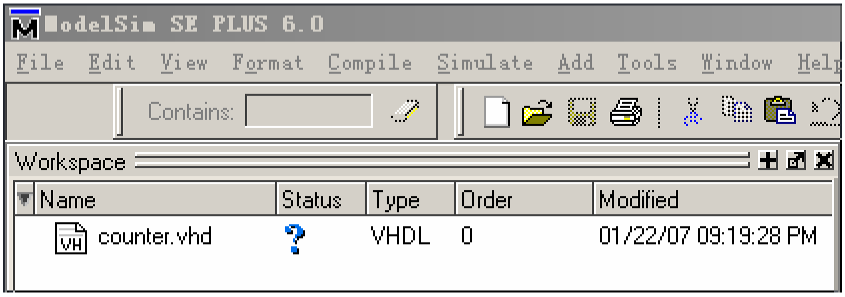
Edit and Compile the Code

To edit VHDL files, you may use the editor provided by ModelSim, or any external code editors such as Windows Notepad, which is normally faster and more stable than ModelSim text editor. ModelSim can also understand DOS and UNIX coded text files.
If you would prefer the VHDL syntax coloring of ModelSim editor, then you can edit/add a VHDL file by choosing:
File -> Add to Project -> New File…, or

File -> Add to Project -> Existing File…
If it is a new file, then create a file name such as “counter.vhd”. Edit it as below:

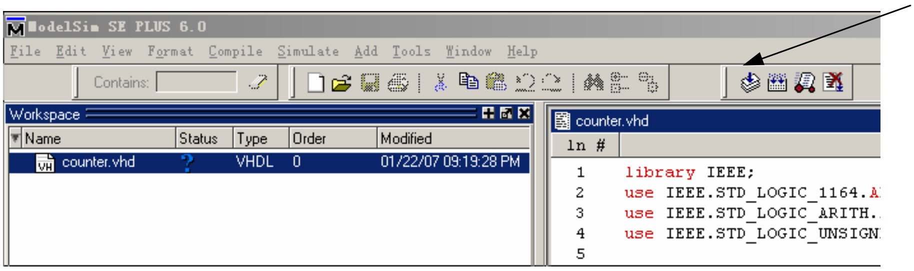
```
entity counter is
   port (count : buffer bit_vector(8 downto 1); -- the most significant bit is
                                                 -- buffer (8)
            clk     : in bit;
            reset : in bit);
end;

architecture only of counter is
     constant tpd_reset_to_count : time := 10 ns;
     constant tpd_clk_to_count      : time :=    5 ns;

     function increment(val : bit_vector) return bit_vector
     is
          -- normalize the indexing
          alias input : bit_vector(val'length downto 1) is val;
          variable result : bit_vector(input'range) := input;
          variable carry : bit := '1';
     begin
          for i in input'low to input'high loop
               result(i) := input(i) xor carry;
               carry := input(i) and carry;
               exit when carry = '0';
          end loop;
          return result;
     end increment;
begin

     ctr:
     process(clk, reset)
     begin
          if (reset = '1') then
               if reset'event then
                    count <= (others => '0') after tpd_reset_to_count;
               end if;
          elsif clk'event and (clk = '1') then
               count <= increment(count) after tpd_clk_to_count;
          end if;
     end process;

end only;
```

After you save it, the workspace window will be like this:


ModelSim SE PLUS 6.0
File Edit View Format Compile Simulate Add Tools Window Help
Contains:
Workspace

Name	Status	Type	Order	Modified
counter.vhd	?	VHDL	0	01/22/07 09:19:28 PM

Highlight the file to activate the “compile” button on the toolbar:

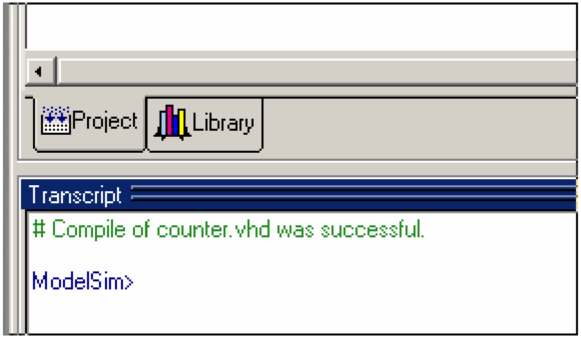

ModelSim SE PLUS 6.0
File Edit View Format Compile Simulate Add Tools Window Help
Contains:
Workspace
counter.vhd
ln #

Name	Status	Type	Order	Modified
counter.vhd	?	VHDL	0	01/22/07 09:19:28 PM

```
library IEEE;
use IEEE.STD_LOGIC_1164.A
use IEEE.STD_LOGIC_ARITH.
use IEEE.STD_LOGIC_UNSIGN

```

“compile”
button

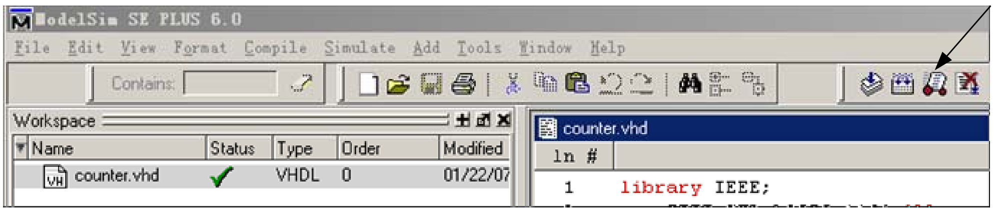
If there is no error in the file, the transcript window at the bottom will show:



```
# Compile of counter.vhd was successful.

ModelSim>
```

● Run the Simulation and View the Waveform

With the file status being changed in the workspace after compilation, you can press the “simulate” button to activate the simulation process.

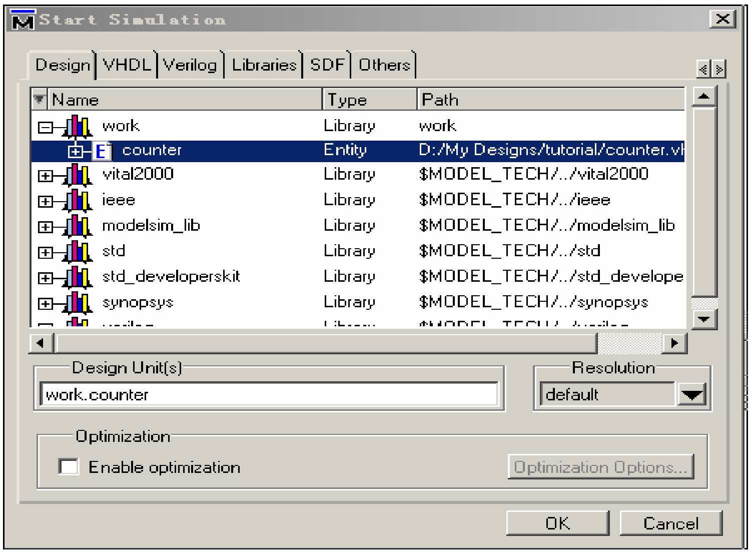

ModelSim SE PLUS 6.0
File Edit View Format Compile Simulate Add Tools Window Help
Contains:
Workspace
counter.vhd
ln #

Name	Status	Type	Order	Modified
counter.vhd	✓	VHDL	0	01/22/07

```
library IEEE;
```

“simulate”
button

This will bring up the following window:



Name	Type	Path
work	Library	work
counter	Entity	D:/My Designs/tutorial/counter.vl
vital2000	Library	$MODEL_TECH/../vital2000
ieee	Library	$MODEL_TECH/../ieee
modelsim_lib	Library	$MODEL_TECH/../modelsim_lib
std	Library	$MODEL_TECH/../std
std_developerskit	Library	$MODEL_TECH/../std_develope
synopsys	Library	$MODEL_TECH/../synopsys

Design Unit(s): work.counter

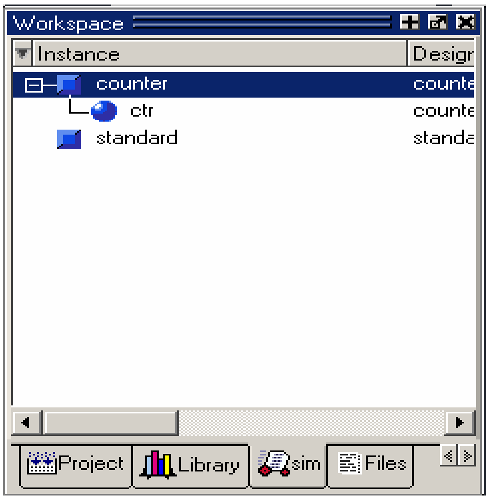
Resolution: default

Optimization

□ Enable optimization

Choose *counter* in the library *work*, and press OK.

Then, the workspace switches to the simulation window by itself:

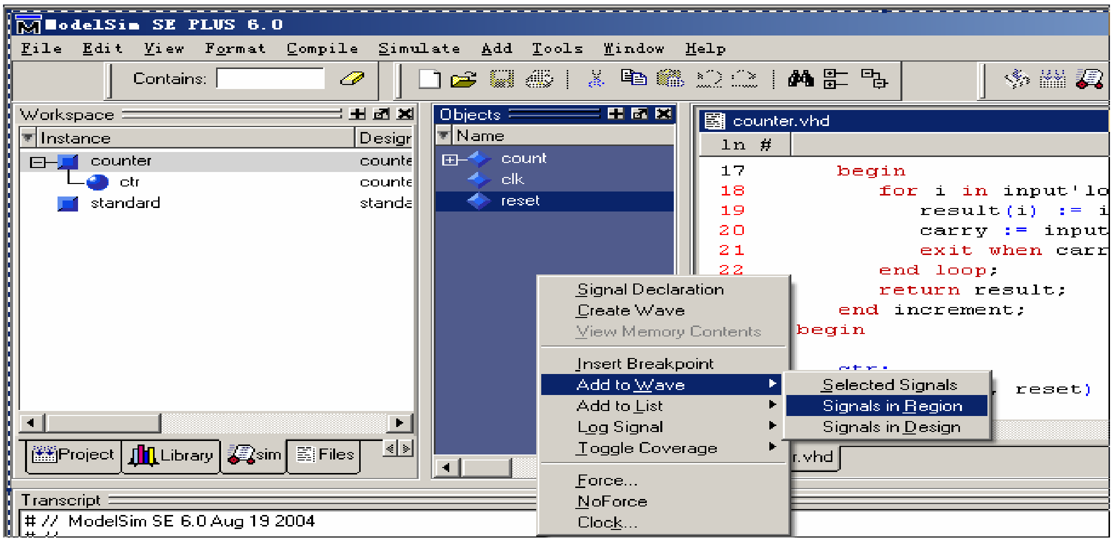


Instance	Desigr
counter	counte
ctr	counte
standard	standa

Now, you can add selected signals to the waveform.

Select View -> Debug Windows -> Objects

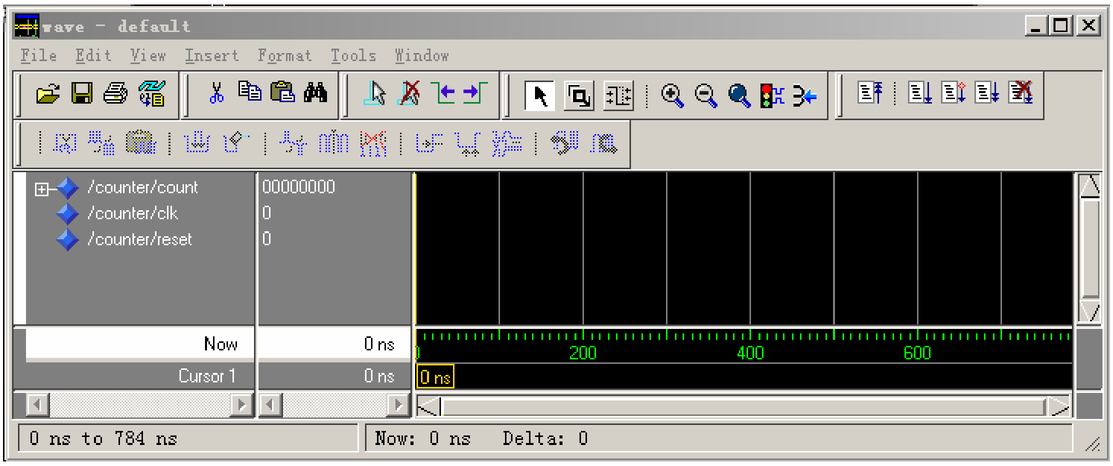
This will bring up a signals window that looks like this:



```
        begin
            for i in input'lo
                result(i)  := i
                carry := input
                exit when carr
            end loop;
            return result;
        end increment;
begin
                                  reset)
```

```
# //  ModelSim SE 6.0 Aug 19 2004
```

After the selected signals or signals in the whole region are added, a default waveform will appear with the signals on the vertical axis and the time along the horizontal axis as shown below:



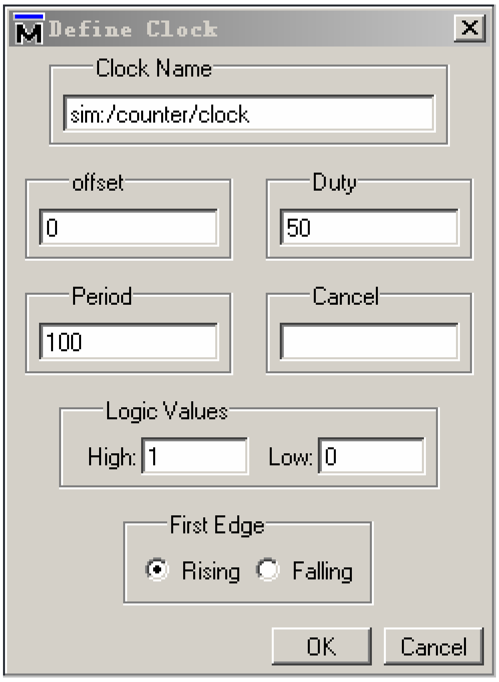
/counter/count 00000000
/counter/clk 0
/counter/reset 0

Now 0 ns
Cursor 1 0 ns

0 ns to 784 ns
Now: 0 ns Delta: 0

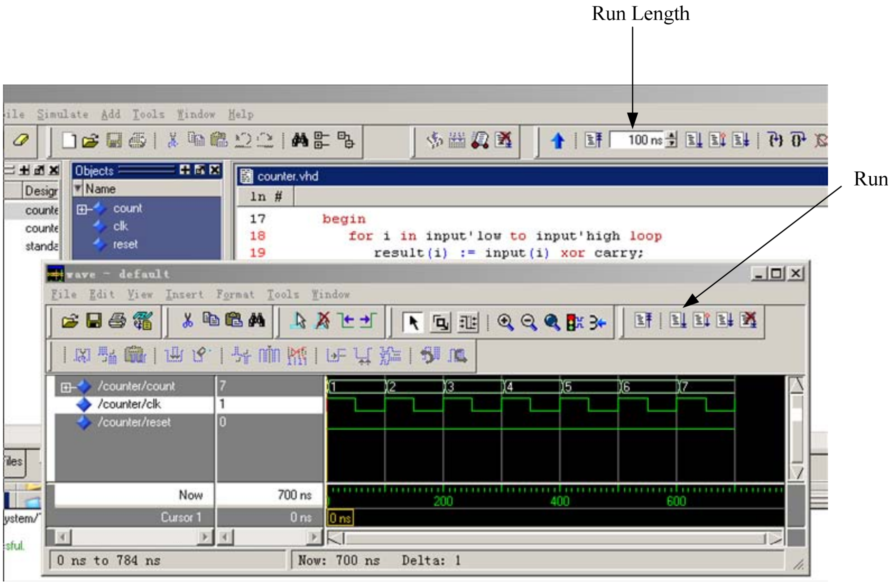
If you right click on the signals, you can change the display radix to make it easier to see whether the counter works correctly. Go to Radix, click on *unsigned*.

It is necessary to set values manually for the input since there is no testbench in this case.
Right click the signal *clock*, select clock, define it by setting the parameters:



Clock Name: sim:/counter/clock
offset: 0
Duty: 50
Period: 100
Cancel:
Logic Values: High: 1 Low: 0
First Edge: ● Rising ○ Falling

Other input signals are assigned differently.
Right click the signal , select *force* and set a specific value in a binary format. Press the *run* button on the toolbar, the waveform looks like the one below:



```
begin
   for i in input'low to input'high loop
      result(i) := input(i) xor carry;
```

The wave that you see above is a box with numbers in it. If you change the radix back to binary, when you see 10000, it means that this is a bus (collection of wires) instead of an individual input or output. You can see the individual wires for each bus if you click “⊞” next to each signal.

The simulation duration can be set by entering a value in the Run Length box.

Summary

In this tutorial, we have built a project for a counter that is edited and simulated for you to get familiar with ModelSim CAD tool. Neither gate delays nor interconnect delays are considered since it is only a behavioral simulation that gives the least accurate prediction of how the circuit will perform. However, it is very useful during the initial debugging of your design.