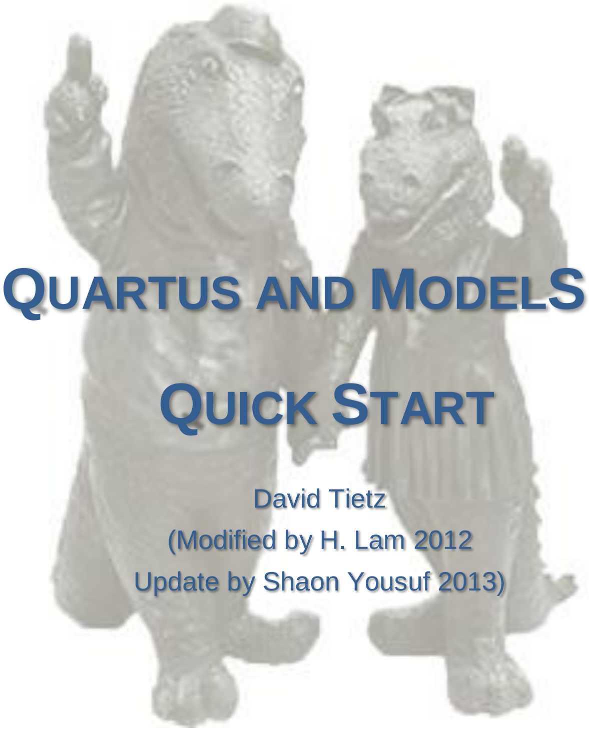


Quartus and ModelSim

Quick Start

David Tietz

(Modified by H. Lam 2012

Update by Shaon Yousuf 2013)

Department of Electrical and Computer Engineering

UF UNIVERSITY of FLORIDA
The Foundation for The Gator Nation

Contents

Installation

1. Download and install the newest version of Quartus II Web Edition Software and ModelSim-Altera Starter Edition Software:

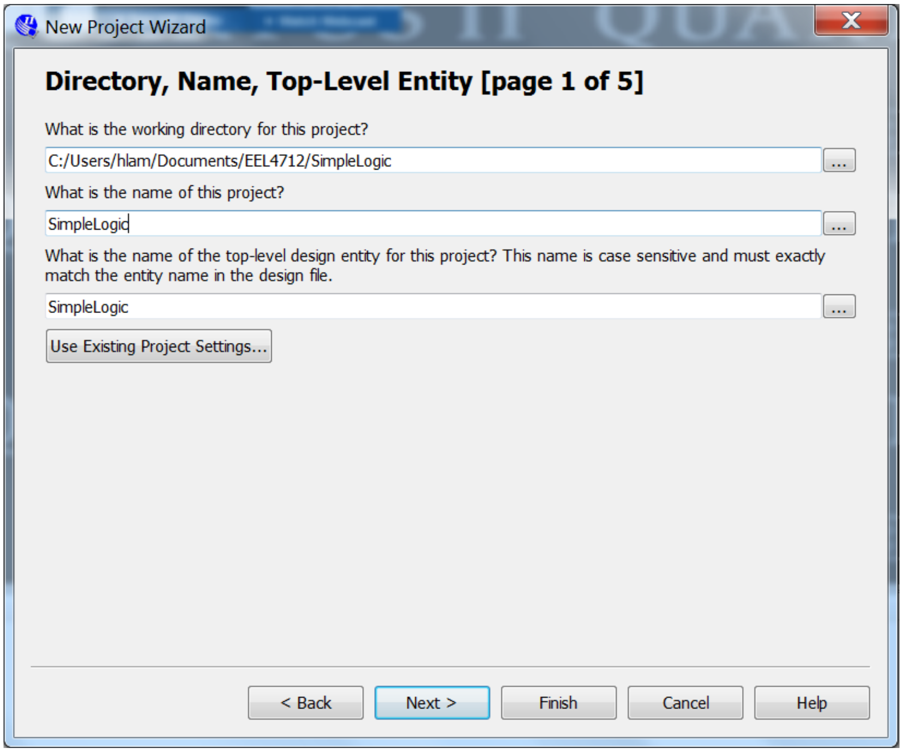
   http://www.altera.com/products/software/quartus-ii/web-edition/qts-we-index.html

   - Be sure the checkbox for Quartus II Programmer and SignalTap II is checked.

Create a Quartus II Project

Once you start Quartus II web edition. You must first create a new project by selection New Project Wizard from the welcome screen:

1. At the Introduction page, click next.
2. Start by creating a new directory and project name, shown in **Figure 1**. Save it somewhere you can access it easily (e.g., …/Documents/EEL4712/SimpleLogic). Then click next.



New Project Wizard

Directory, Name, Top-Level Entity [page 1 of 5]

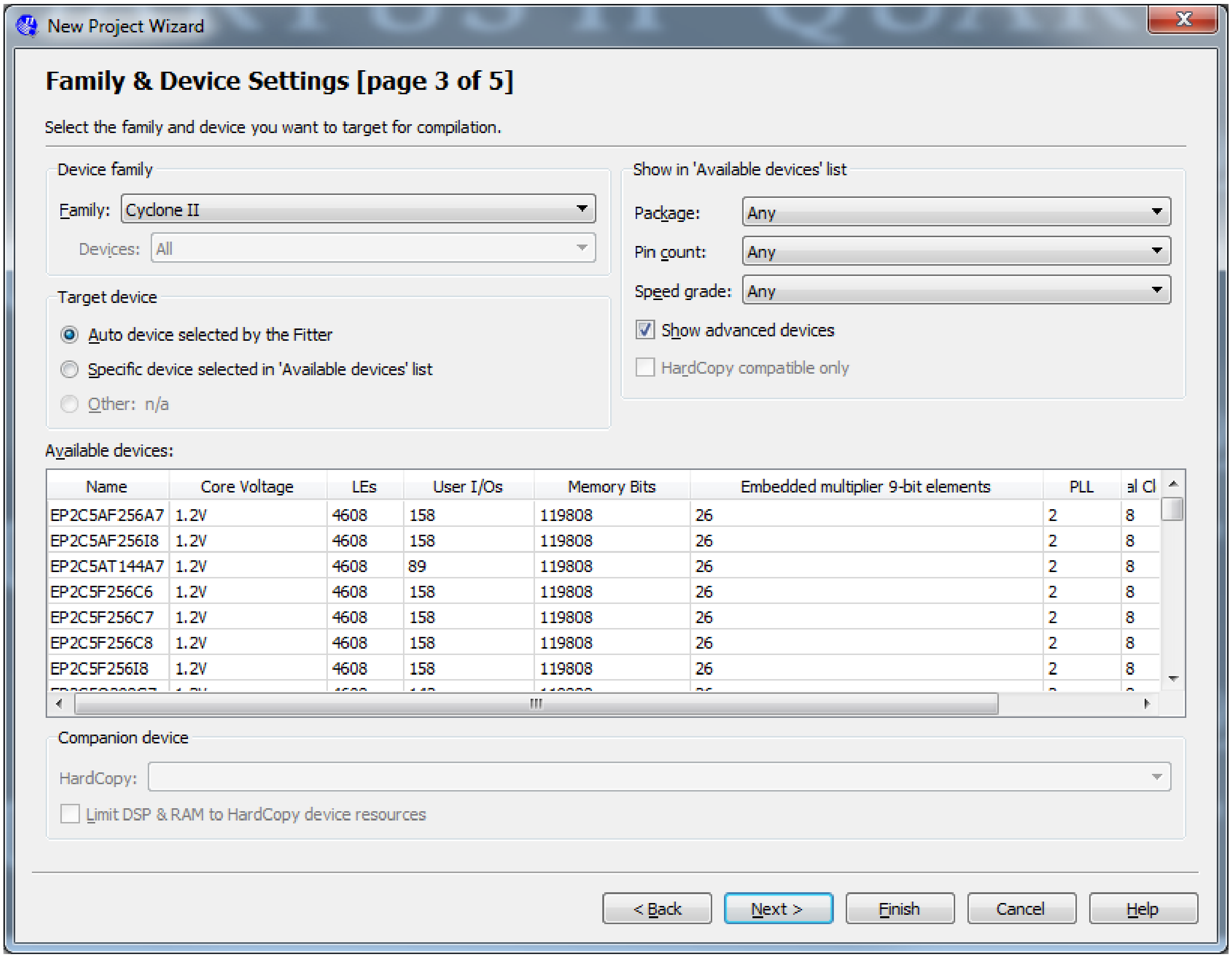
What is the working directory for this project?
C:/Users/hlam/Documents/EEL4712/SimpleLogic

What is the name of this project?
SimpleLogic

What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.
SimpleLogic

Figure 1

3. In the next page, you would normally add design files to your project, if this is your first lab, you don't have any to import, so just click next.
4. In the family and device setting step, select 'Auto device selected by the Fitter' or select the correct device we will be using for EEL 4712, shown in **Figure 2** then click next.



New Project Wizard

Family & Device Settings [page 3 of 5]

Select the family and device you want to target for compilation.

Device family

Family: Cyclone II

Devices: All

Target device

● Auto device selected by the Fitter

○ Specific device selected in 'Available devices' list

○ Other: n/a

Show in 'Available devices' list

Package: Any

Pin count: Any

Speed grade: Any

☑ Show advanced devices

☐ HardCopy compatible only

Available devices:

Name	Core Voltage	LEs	User I/Os	Memory Bits	Embedded multiplier 9-bit elements	PLL	al Cl
EP2C5AF256A7	1.2V	4608	158	119808	26	2	8
EP2C5AF256I8	1.2V	4608	158	119808	26	2	8
EP2C5AT144A7	1.2V	4608	89	119808	26	2	8
EP2C5F256C6	1.2V	4608	158	119808	26	2	8
EP2C5F256C7	1.2V	4608	158	119808	26	2	8
EP2C5F256C8	1.2V	4608	158	119808	26	2	8
EP2C5F256I8	1.2V	4608	158	119808	26	2	8

Companion device

HardCopy:

☐ Limit DSP & RAM to HardCopy device resources

Figure 2

5. In EDA Tool Settings, select the following (i.e., ModelSim-Altera) from the pull down menus, shown in **Table 1**, and click next:

Tool Type	Tool Name
Design Entry/Synthesis	<None>
Simulation	**ModelSim-Altera**
Timing Analysis	<None>
Format Verification	<None>

Table 1

6. On the summary page, click finish.

Adding Design Files

1. (a) From the file menu, select new. From the Menu, select the type of design file you would like to create (in this case VHDL), then click OK, as shown in **Figure 3**.

Type in the VHDL code from SimpleLogic.vhd

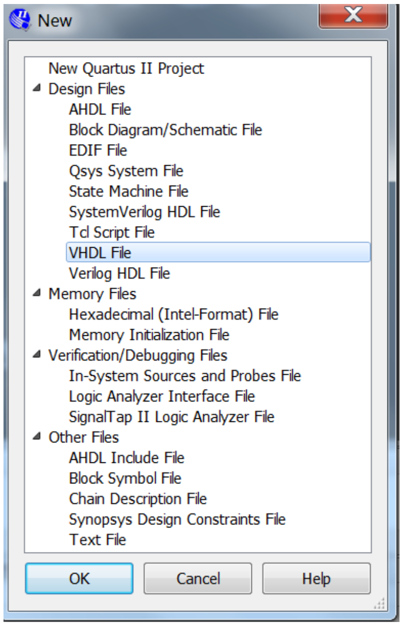


Figure 3

1. (b) Alternatively, you can add the file (SimpleLogic.vhd) to the project (without having to type it in:

   Click Project→Add/Remove File from Project …
   Browse to the directory and select SimpleLogic.vhd
   Click **Add** and then **OK**

2. If necessary, confirm that the design file was added to the current project, by click from the top menu: Project →Add Current File to Project.

Setting Up the EDA Simulator Execution Path

To run an EDA simulator (e.g. Modelsim-Altera) automatically from the Quartus II software using the NativeLink feature, specify the path to your simulation tool by performing the following steps:

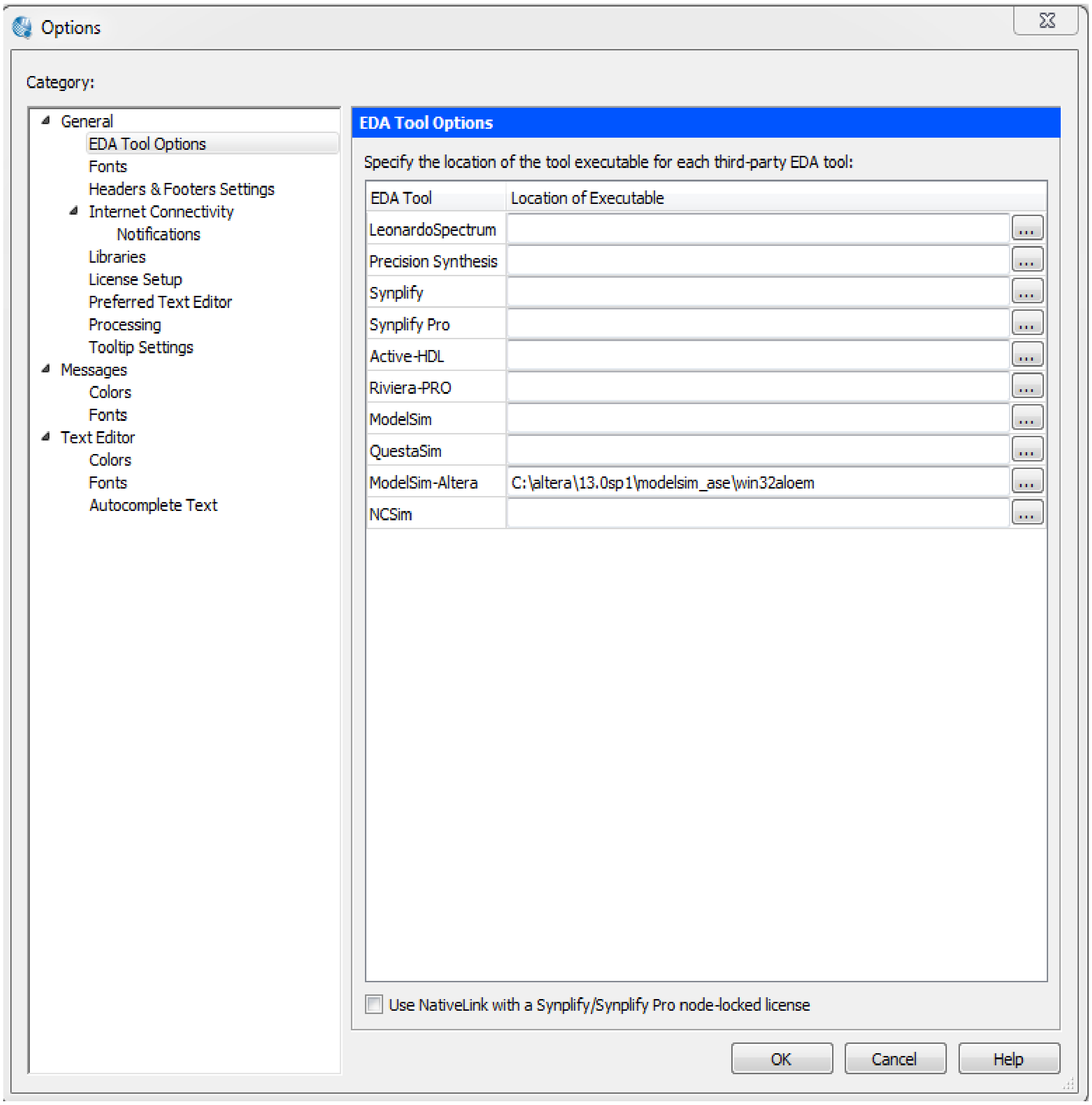
1. On the Tools menu, click **Options**. The **Options** dialog box appears.
2. In the **Category** list, under the **General** category, select **EDA Tool Options**.
3. The **Options** window should look like Figure 4. In the **Modelsim-Altera** entry, the location of executable should be something like "C:\altera\13.0sp1\modelsim_ase\win32aloem".
4. If not, then browse to the directory containing the executable of the Modelsim-Altera simulator. (Again, path should be like "C:\altera\13.0sp1\modelsim_ase\win32aloem".)
5. Click **OK**.

Compiling the Design

Once you have all the design files you need in place, you are ready to compile:

1. On the top menu, select: Processing → Start Compilation.

2. The message window below will show compilation status, and you will also be shown a compilation report window. If the design compiles successfully, you can close out the report.



Specify the location of the tool executable for each third-party EDA tool:

EDA Tool	Location of Executable
LeonardoSpectrum	
Precision Synthesis	
Synplify	
Synplify Pro	
Active-HDL	
Riviera-PRO	
ModelSim	
QuestaSim	
ModelSim-Altera	C:\altera\13.0sp1\modelsim_ase\win32aloem
NCSim	

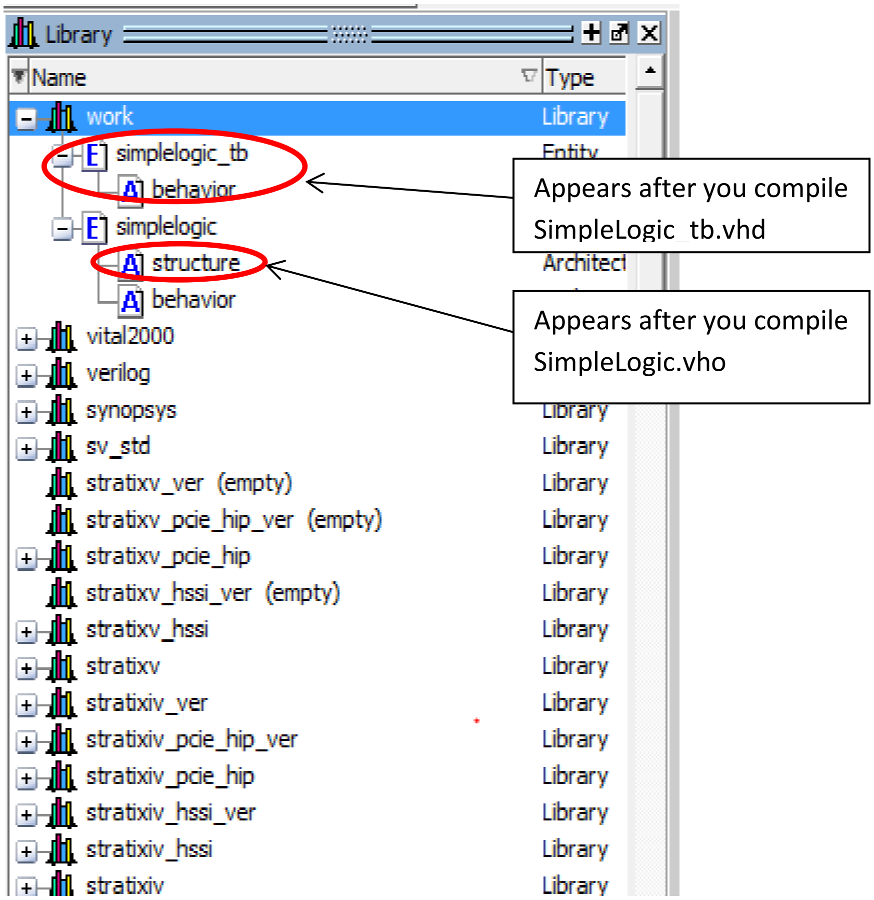
☐ Use NativeLink with a Synplify/Synplify Pro node-locked license

Figure 4

Compilation in ModelSim-Altera

Now that you have created and compiled the project in Quartus II. You are ready to use ModelSim to perform the testbench simulations, but first you need to compile your design files in ModelSim

1. Invoke ModelSim from Quartus: Tools → Run Simulation Tool → RTL Simulation
2. On ModelSim open the compile window by clicking Compile → Compile
3. Compile the following files:
    a. SimpleLogic.vho (should be in the simulation/modelsim directory)
    b. SimpleLogic_tb.vhd (Provided to you in Lab1vhdlFiles.zip and placed by you in some directory)
4. Click **Done** and open the library view if it's not already open (by selecting View → Library) and verify the red circled portions as shown in Figure 5 are there.
5. If there are errors, try compiling again because an incorrect compilation order can cause this problem. If there are errors in the testbench, you likely didn't name your I/O correctly in your Quartus design file (SimpleLogic.vhd).



Name	Type
work	Library
simplelogic_tb	Entity
behavior	
simplelogic	
structure	Architect
behavior	
vital2000	
verilog	
synopsys	Library
sv_std	Library
stratixv_ver (empty)	Library
stratixv_pcie_hip_ver (empty)	Library
stratixv_pcie_hip	Library
stratixv_hssi_ver (empty)	Library
stratixv_hssi	Library
stratixv	Library
stratixiv_ver	Library
stratixiv_pcie_hip_ver	Library
stratixiv_pcie_hip	Library
stratixiv_hssi_ver	Library
stratixiv_hssi	Library
stratixiv	Library

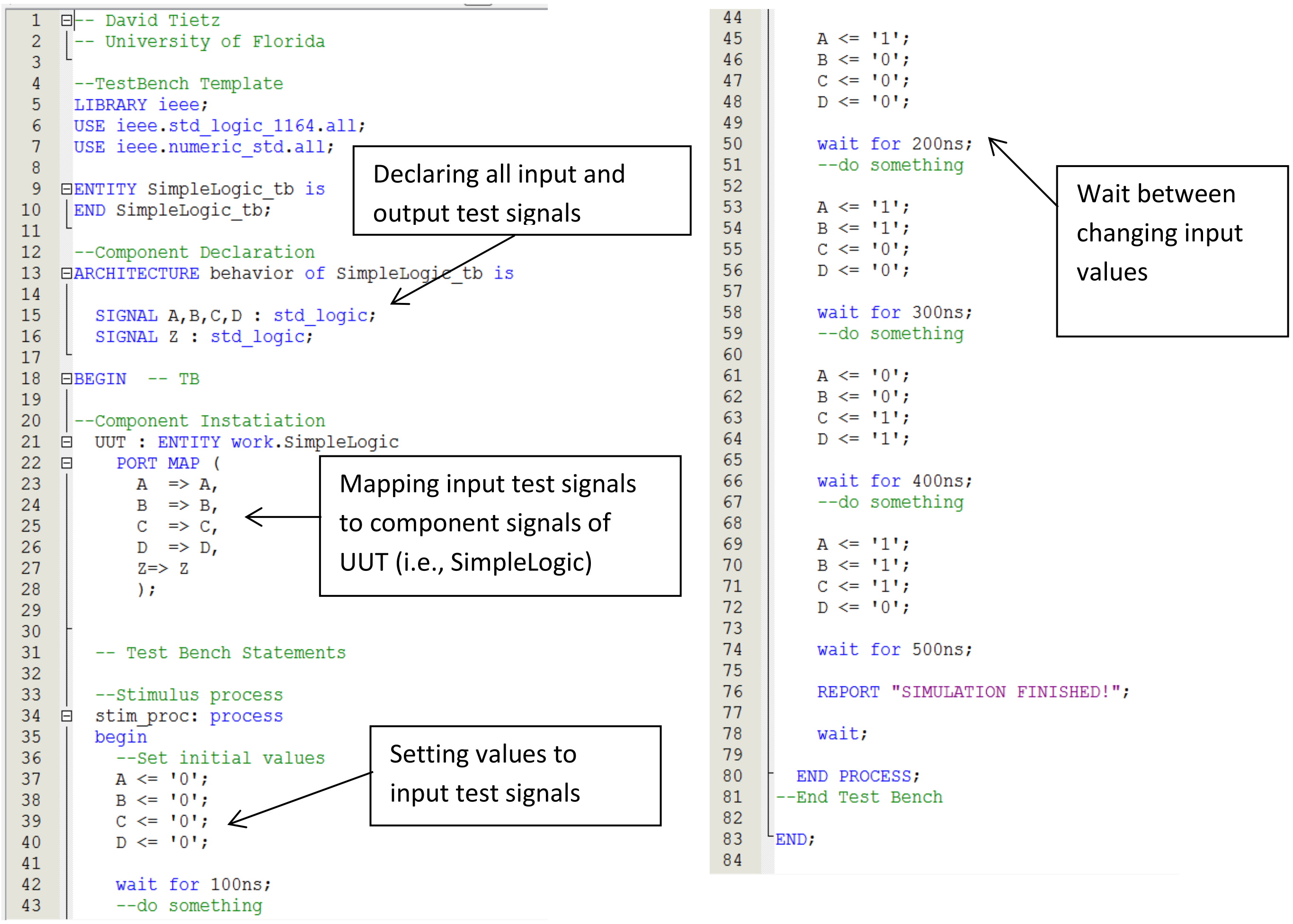
Appears after you compile SimpleLogic_tb.vhd

Appears after you compile SimpleLogic.vho

Figure 5

Understanding a Testbench File

Testbench files are used to test your design files as against a set of input test signals. Input test signals are generated and applied to the unit under test (UUT) within the test bench. **Figure 6** is a testbench file we used for this tutorial.



```
-- David Tietz
-- University of Florida

--TestBench Template
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

ENTITY SimpleLogic_tb is
END SimpleLogic_tb;

--Component Declaration
ARCHITECTURE behavior of SimpleLogic_tb is

 SIGNAL A,B,C,D : std_logic;
 SIGNAL Z : std_logic;

BEGIN  -- TB

--Component Instatiation
 UUT : ENTITY work.SimpleLogic
  PORT MAP (
   A  => A,
   B  => B,
   C  => C,
   D  => D,
   Z=> Z
   );


 -- Test Bench Statements

 --Stimulus process
 stim_proc: process
 begin
  --Set initial values
  A <= '0';
  B <= '0';
  C <= '0';
  D <= '0';

  wait for 100ns;
  --do something

  A <= '1';
  B <= '0';
  C <= '0';
  D <= '0';

  wait for 200ns;
  --do something

  A <= '1';
  B <= '1';
  C <= '0';
  D <= '0';

  wait for 300ns;
  --do something

  A <= '0';
  B <= '0';
  C <= '1';
  D <= '1';

  wait for 400ns;
  --do something

  A <= '1';
  B <= '1';
  C <= '1';
  D <= '0';

  wait for 500ns;

  REPORT "SIMULATION FINISHED!";

  wait;

 END PROCESS;
--End Test Bench

END;

```

Declaring all input and output test signals

Mapping input test signals to component signals of UUT (i.e., SimpleLogic)

Setting values to input test signals

Wait between changing input values

Figure 6

For future designs, you will need to make or modify the above testbench file to fit the needs of the design you are trying to simulate. You will need to declare the test signals you will use, port map them to correct component signals from the Quartus design, and then setup the actual signal transitions.

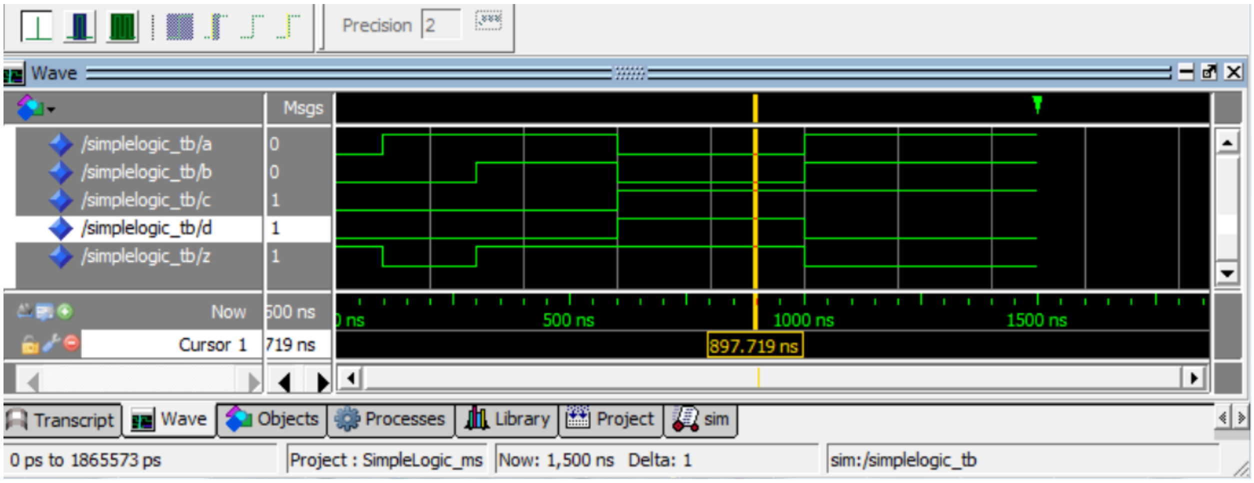
Running a Functional Simulation

In a functional simulation, you will simulate your design based on a functional stand point. In other words, delays through the system will not be taken into account. To run a functional simulation:

1. Click Simulate → Start Simulation.
2. On the Design tab select work →SimpleLogic_tb, which is the testbench. Click OK.
3. Two simulation windows should load: **Object** window and **Wave** window. If not, you can always load them yourself by clicking the **View** menu and checking the appropriate window.
4. Generally you want to display all the test signals in the Wave window. To do so, click in the **Objects** window
   - Click Add -> To Wave -> Signals in Region

   Alternatively, you can display only selected signals in the Wave window:

   - In the Objects window, select the signals you would like to monitor and drag them into the Wave window (or copy and paste).
5. Click Simulate → Run → Run -All.
6. You can use 'Ctrl +' and 'Ctrl -' to zoom in and out. **Figure 7** shows the results of the simulation.



0 ps to 1865573 ps | Project : SimpleLogic_ms | Now: 1,500 ns Delta: 1 | sim:/simplelogic_tb

Figure 7

Running a Timing Simulation

In a timing simulation, you will simulate your design based on a timing stand point. In other words, delays through the system will be included and will affect the performance of your design. To run a timing simulation:

1. Make sure the simulation currently running is complete by clicking Simulation→End Simulation.
2. Quartus creates an .sdo file to annotate simulation with actual propagation delays. To include this in your ModelSim project:
   a. Select the SDF tab (on the dialog box that comes up after Start Simulation).
   b. Add the SimpleLogic.sdo file that is in the quartus_project/simulation/modelsim directory.
   c. This should be same directory as the .vho file that you previously added.
   d. In the “Apply to Region” text box, make sure to type **/UUT** (this is the same label used in the testbench file for that region of component instantiation).
3. On the Design tab select work->SimpleLogic_tb, which is the testbench. Click OK. Three simulation screens will load.
4. In the Objects window, select the signals you would like to monitor and drag them into the Wave view.
5. Click Simulate → Run → Run -All.
6. **Figure 8** shows the results of the timing simulation.

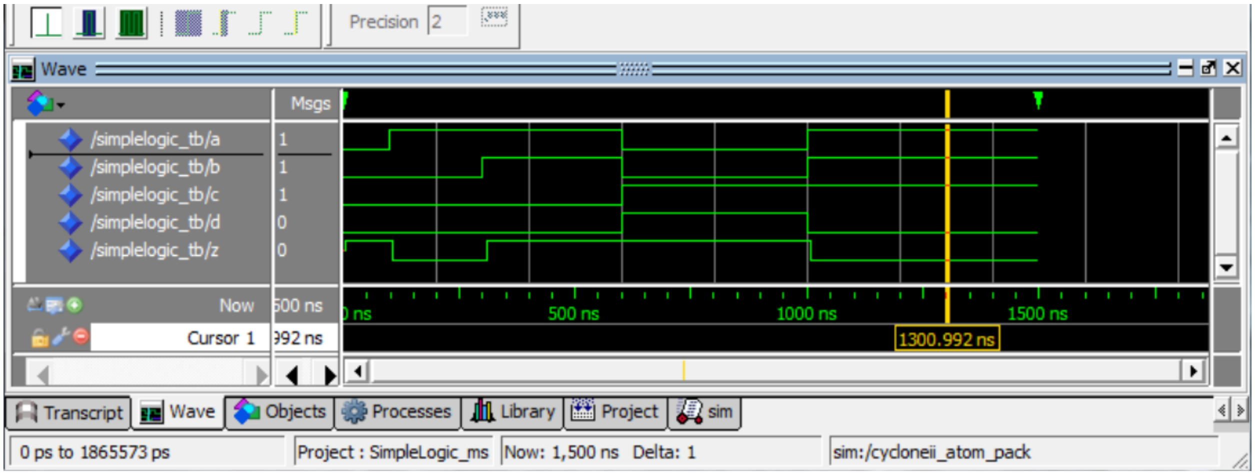


Figure 8