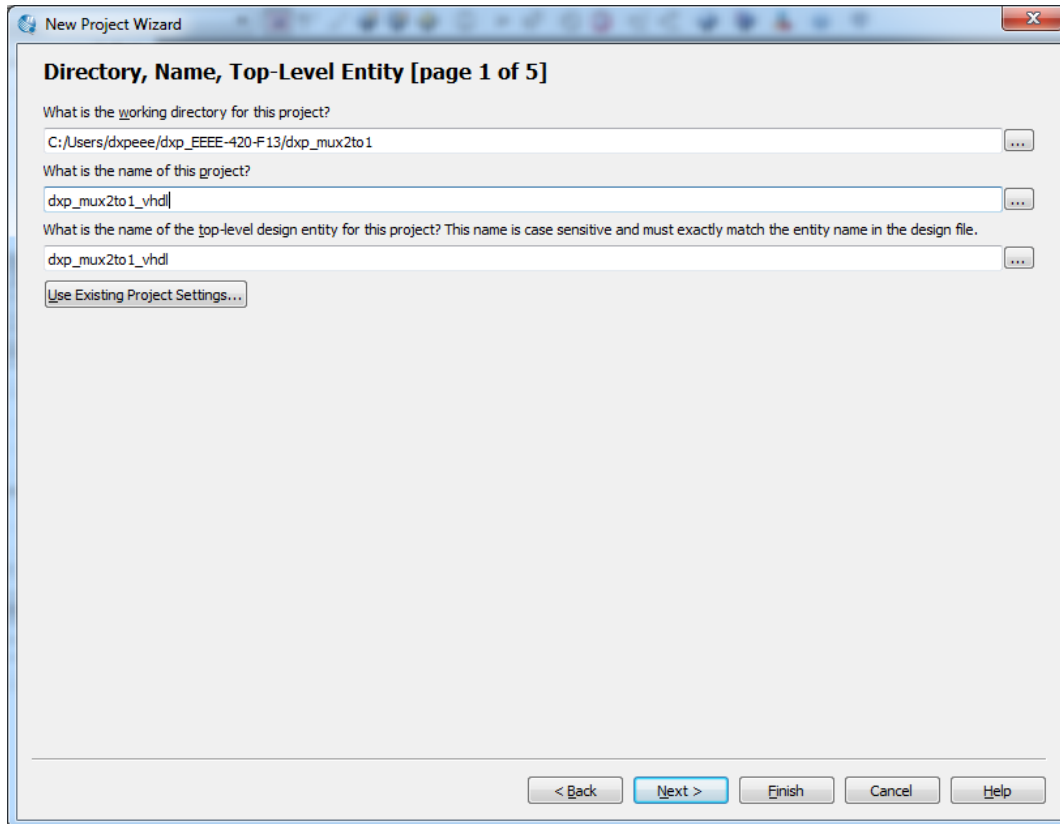


EEEE-220-Lab2

- 1) From mycourses download on your desktop this pdf file. This lab consists of four parts:
 - a. 2-to-1 multiplexer implementation in VHDL,
 - b. 2-to-1 multiplexer implementation in Verilog,
- 2) Start by creating a new project in your working directory that you have created last time: fml_EEEE-220_F13. The **new directory and project names** are fml_mux2to1_vhdl. In all screen capture you'll see instead of fml my initials dxp.



- 3) Click Next twice. In the 3rd step select the proper device:

New Project Wizard

Family & Device Settings [page 3 of 5]

Select the family and device you want to target for compilation.
You can install additional device support with the Install Devices command on the Tools menu.

Device family

Family: **Cyclone IV E**

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**

Name filter:

☒ Show advanced devices ☐ HardCopy compatible only

Available devices:

Name	Core Voltage	LEs	User I/Os	Memory Bits	Embedded multiplier 9-bit elements	PLL	GL
EP4CE22E22...	1.0V	22320	80	608256	132	4	20
EP4CE22E2217	1.2V	22320	80	608256	132	4	20
EP4CE22E22...	1.0V	22320	80	608256	132	4	20
EP4CE22F17...	1.2V	22320	154	608256	132	4	20
EP4CE22F17...	1.2V	22320	154	608256	132	4	20
EP4CE22F17...	1.2V	22320	154	608256	132	4	20
EP4CE22F17...	1.2V	22320	154	608256	132	4	20
EP4CE22F17...	1.2V	22320	154	608256	132	4	20

Companion device

HardCopy:

☐ Limit DSP & RAM to HardCopy device resources

< Back **Next >** Finish Cancel Help

- 4) In step 4 of 5 choose to simulate the tool ModelSim-Altera. The next tab Format(s) will automatically be set to VHDL.

New Project Wizard

EDA Tool Settings [page 4 of 5]

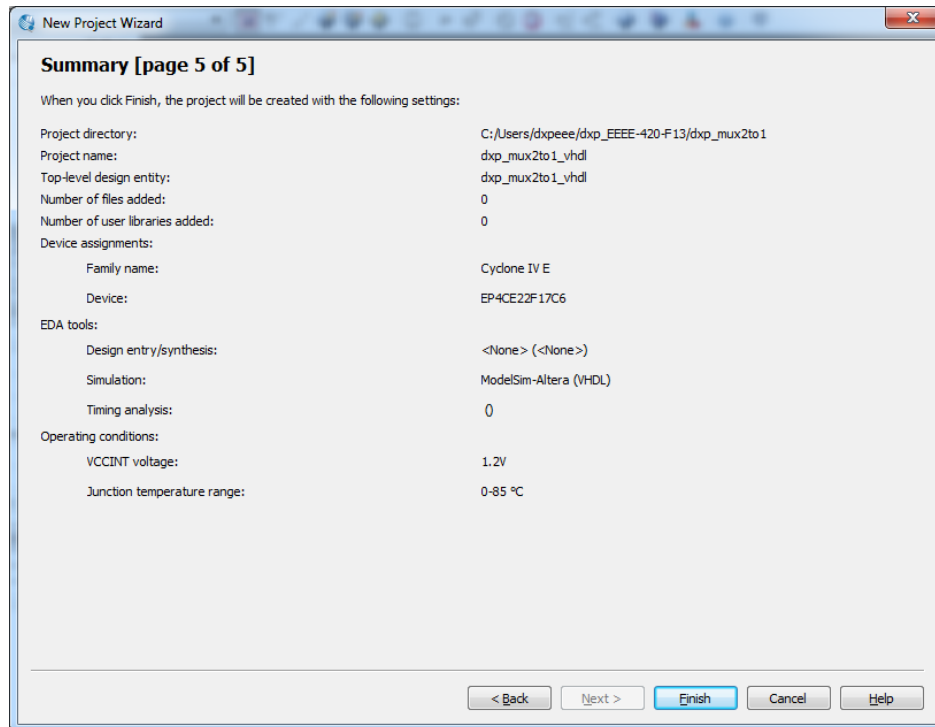
Specify the other EDA tools used with the Quartus II software to develop your project.

EDA tools:

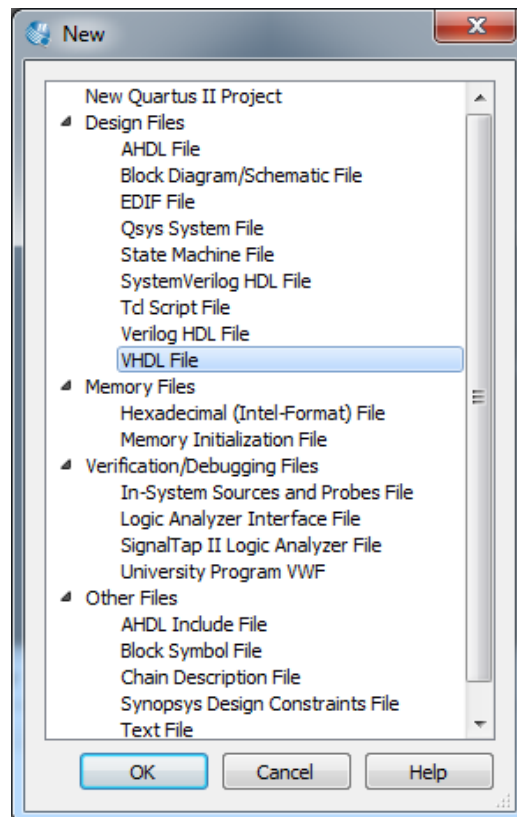
Tool Type	Tool Name	Format(s)	Run Tool Automatically
Design Entry/Synthesis	<None>	<None>	<input type="checkbox"/> Run this tool automatically to synthesize the current design
Simulation	<None>	<None>	<input type="checkbox"/> Run gate-level simulation automatically after compilation
Formal Verification	<None>	<None>	
Board-Level	<None>	<None>	
	Active-HDL	<None>	
	Riviera-PRO	<None>	
	ModelSim	<None>	
	ModelSim-Altera	<None>	
	QuestaSim	<None>	
	NCSim	<None>	
	VCS	<None>	
	VCS MX	<None>	
	Custom	<None>	

< Back **Next >** Finish Cancel Help

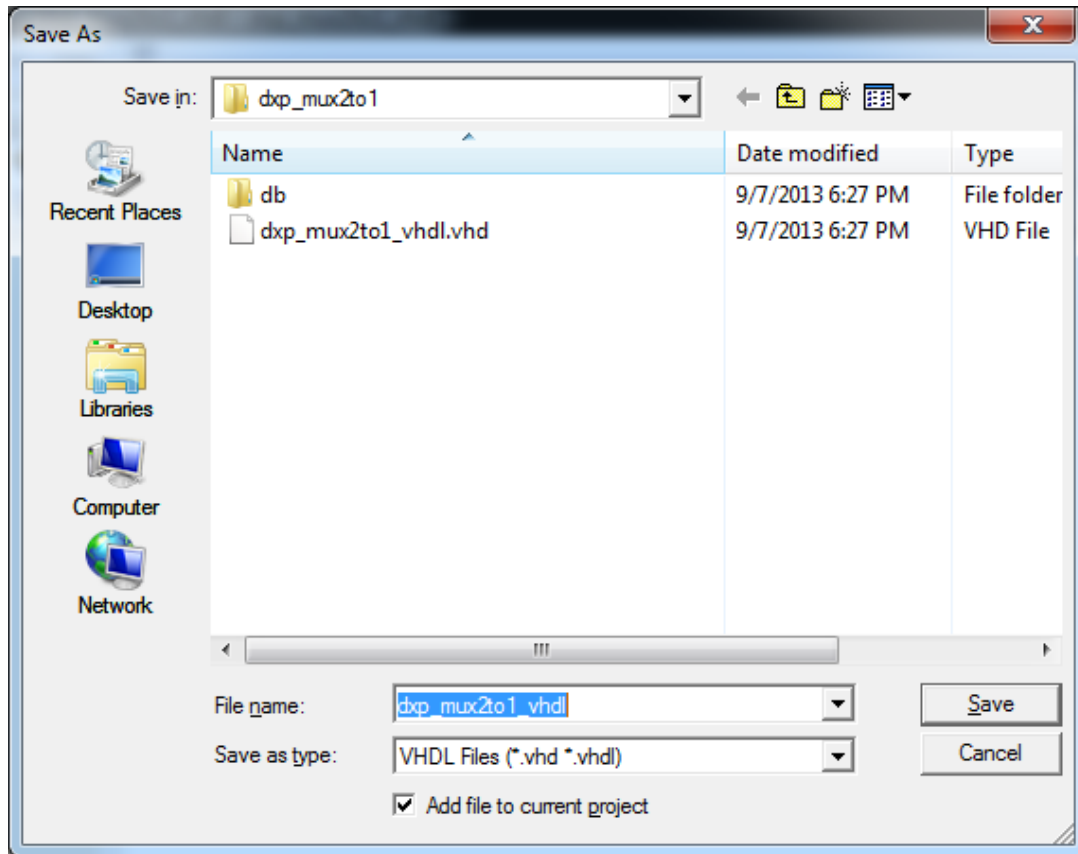
- 5) Click Next and after you check your Summary to match the one below, click Finish.



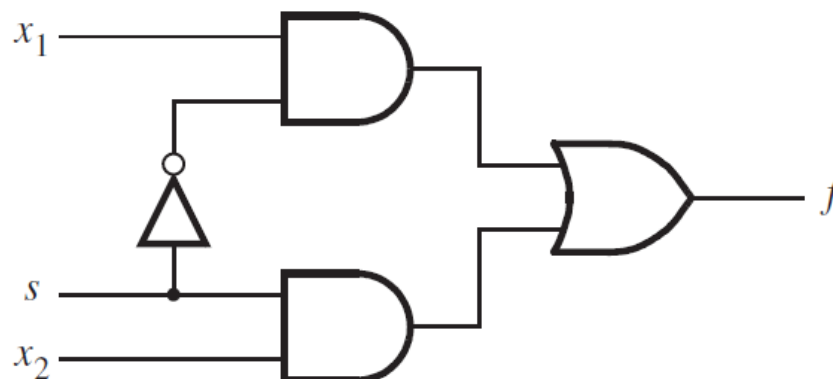
- 6) Next you create the top level design file. Choose File > New and select VHDL file.



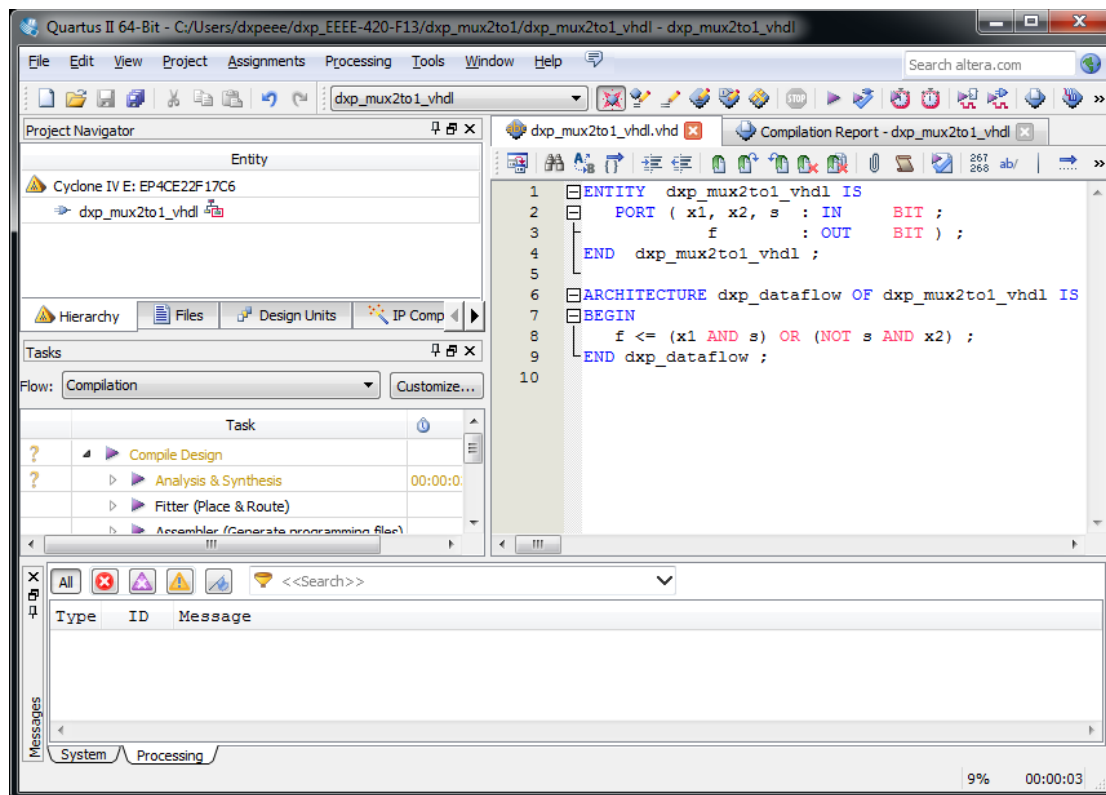
- 7) Click File > Save as and leave the default name and extension. Notice that the default name is the same as your project name. Leave the Add File to Project checked to.



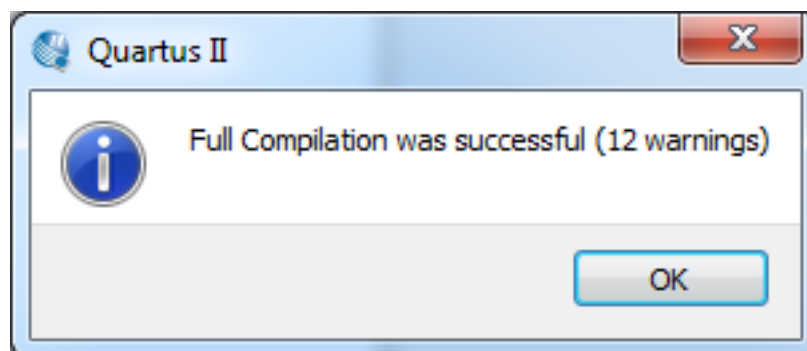
- 8) For your review, the circuit you have to design, simulate, and implement is a 2-to-1 multiplexer:



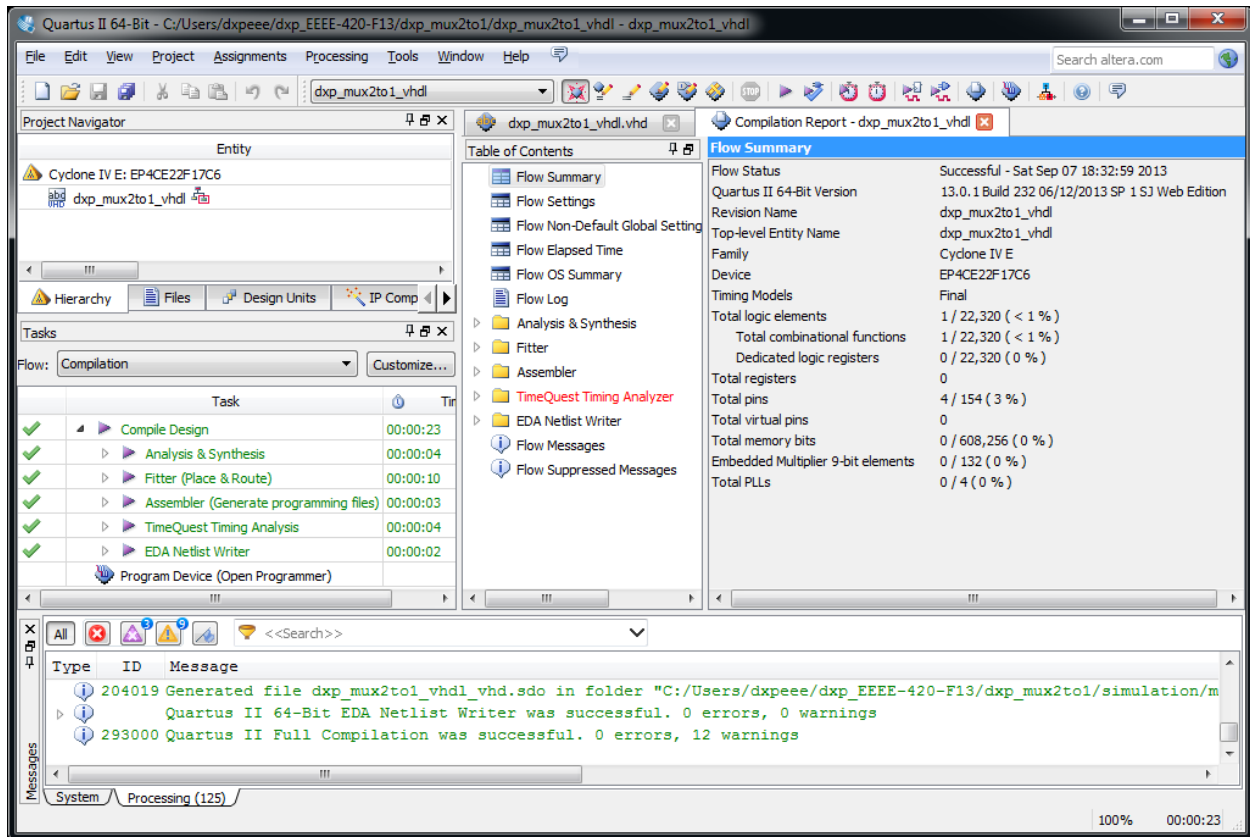
- 9) As is shown in lecture 5, a circuit is described/modeled in VHDL by the use of two major constructs: an ENTITY and an ARCHITECTURE. Type the code below into your VHDL top level design file.



- 10) Notice the fact that the editor color codes VHDL keywords. It also allows you to indent code to make it more readable and collapse/un-collapse code to make it easier to scroll through a large code file. This code has been explained in lecture 5. If needed, open those lecture notes and spend the time to review the significance of every line of code before you proceed to the next steps. Click Save.
- 11) Next we want to compile/synthesize the design. You can start this process by doing one of the following:
 - a. Click Ctrl-L
 - b. Click the magenta play button in the upper toolbar
 - c. Or go to Processing > Start Compilation
- 12) If all is going well you will see the window below:



13) After you click OK, you'll see a report as shown below. If you have errors you must go back to your code and fix them.



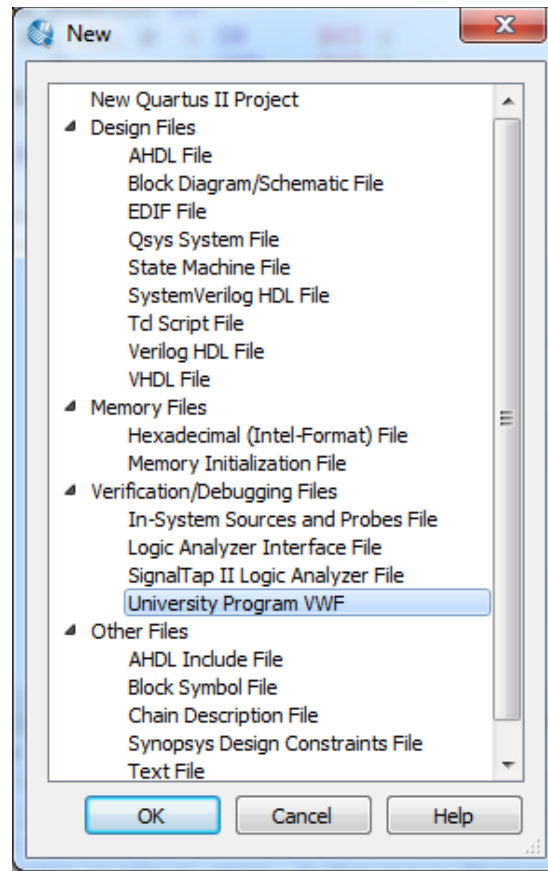
14) Let's take some time and review the information in this window. In the left middle tab you see that the "Compile Design" in Altera Quartus II is actually a sequence of several processing steps.

- During "Analysis and Synthesis" the design is checked for syntactic and semantic (meaning) errors, and then synthesized to a structural gate-level description.
- During "Fitter (Place & Route)" the circuit is fitted into the available hardware in the chosen target device.
- The "Assembler" generates the binary programming file that will be downloaded to the device to configure it. More about the other two later on.
- The second column contains information about how long it took each step to complete.

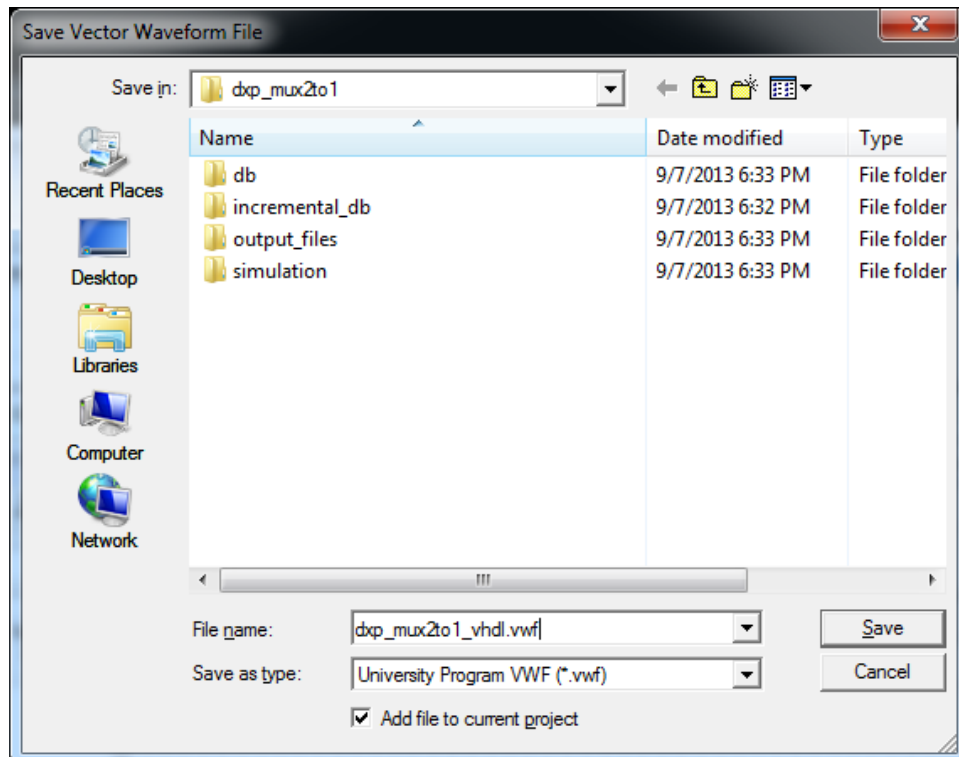
15) At this point you can close the compilation report tab.

16) We now want to proceed to simulation. We need to create first a waveform file and manually setup the stimuli.

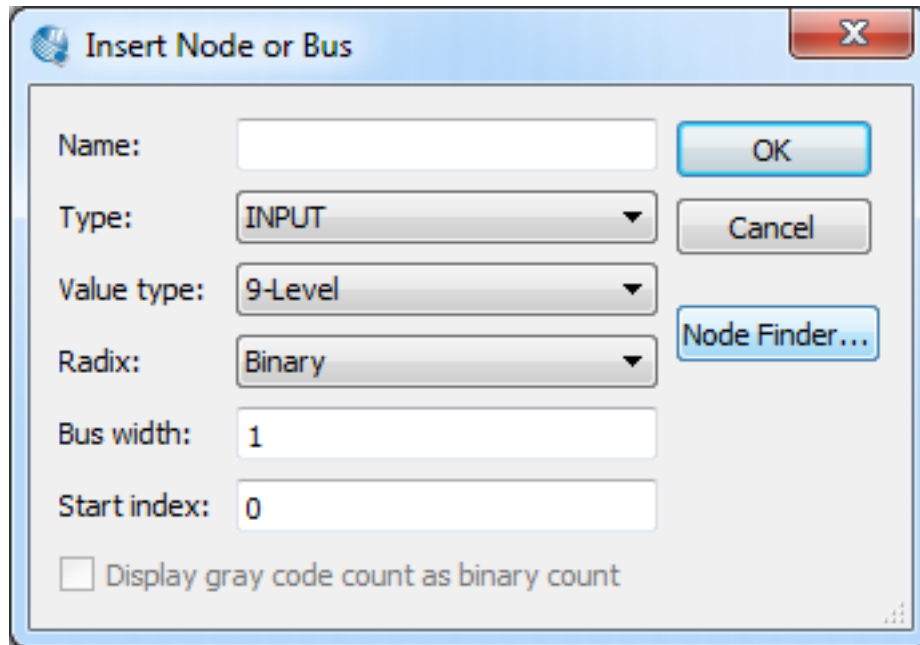
17) Go to File > New and choose "University Program VWF"



18) Save the new file as fml_mux2to1_vhdl.vwf



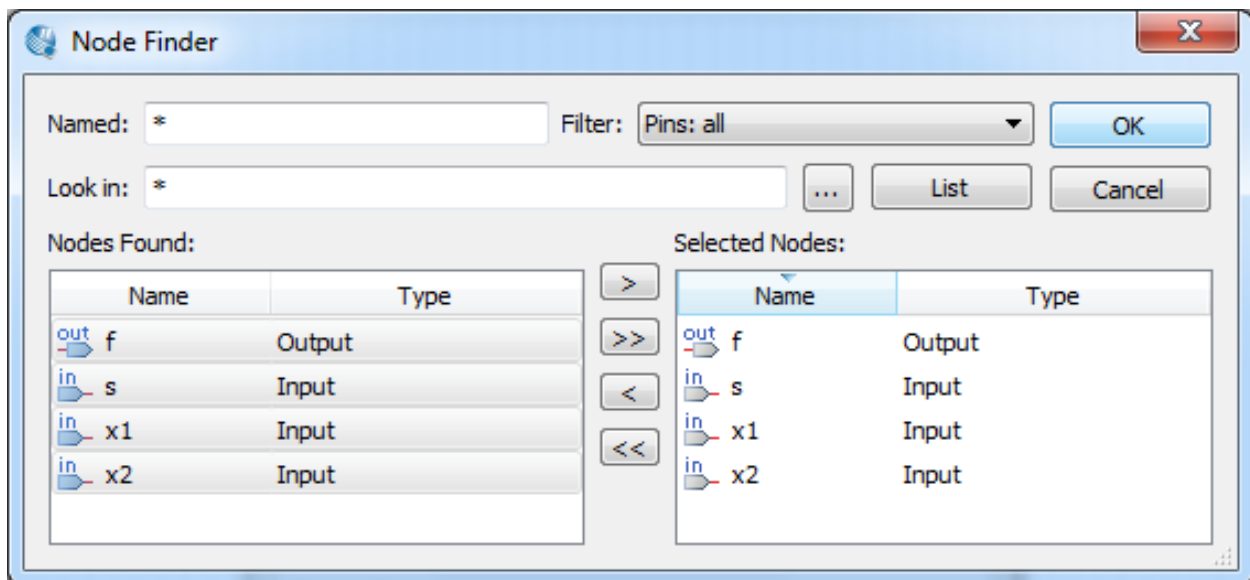
19) Now right click into the LHS column and choose Insert Node or Bus.



The "Insert Node or Bus" dialog box is shown. It has a title bar with a globe icon and a close button. The dialog contains the following fields and buttons:

- Name:
- Type:
- Value type:
- Radix:
- Bus width:
- Start index:
- ☐ Display gray code count as binary count
- Buttons: OK, Cancel, Node Finder...

20) Click on Node Finder and then List. This will list all available pins to include in the simulation waveform file. Click >> to select all. Finish by clicking OK.



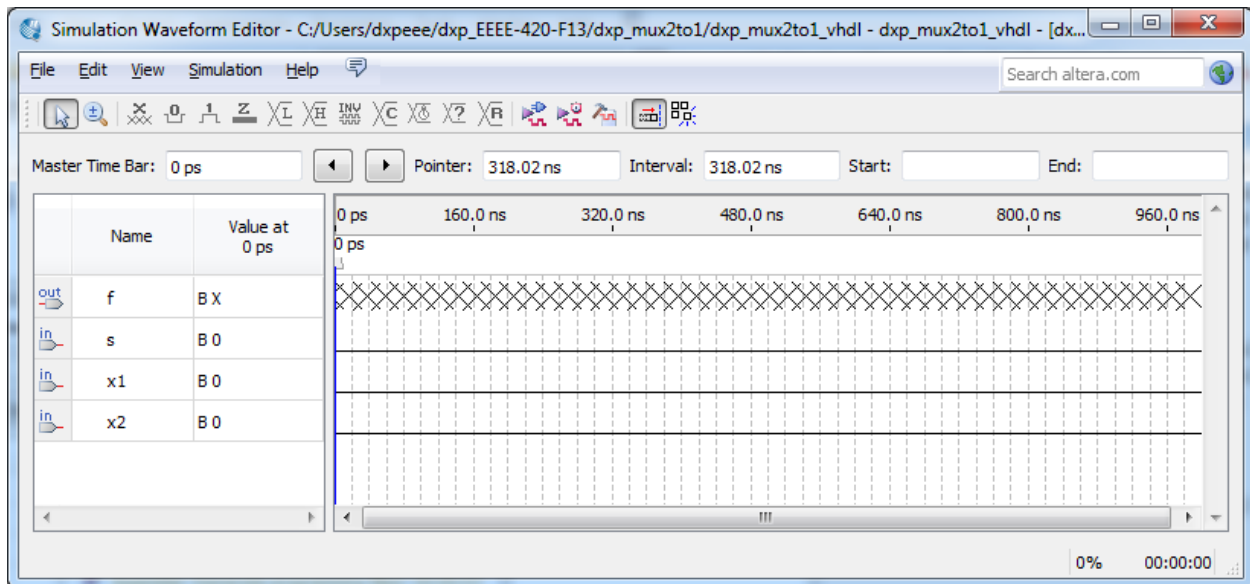
The "Node Finder" dialog box is shown. It has a title bar with a globe icon and a close button. The dialog contains the following fields and buttons:

- Named:
- Filter:
- Look in:
- Buttons: OK, Cancel, List, ...
- Nodes Found:

Name	Type
out f	Output
in s	Input
in x1	Input
in x2	Input
- Selected Nodes:

Name	Type
out f	Output
in s	Input
in x1	Input
in x2	Input
- Navigation buttons: >, >>, <, <<

21) Your waveform window will look like this at this point:



- 22) Notice that the value of the output *f* is undetermined until after the simulation. The three inputs are all by default low or 0 logic. We intend to verify the functionality for all possible input combinations. Instead of manually setting the input waveforms to high and low for 23=8 combinations, we will choose each to change like a clock signal with different frequencies.
- 23) Let's start with the select input port. Select it and then in the upper tab click on the C icon. A window as the one below will open:

Count Value

Radix: **Binary**

Start value: **0**

Increment by: **1**

Count type

☒ Binary

☐ Gray code

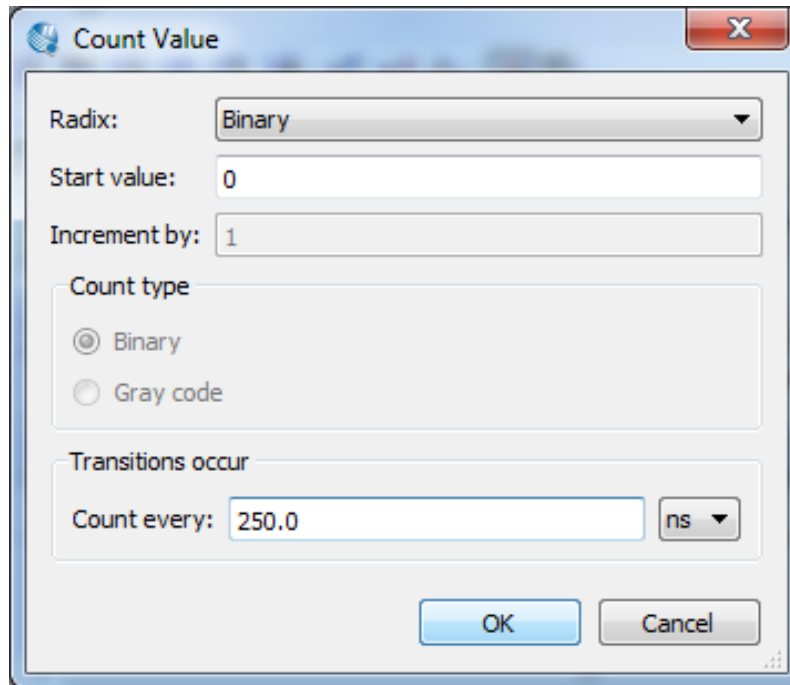
Transitions occur

Count every: **500.0** ns

OK Cancel

- 24) Choose to count 500 ns and then click OK.

25) Do the same for x1, but choose to count every 250 ns:



The 'Count Value' dialog box for x1 is shown. It has a title bar with a close button. The 'Radix' is set to 'Binary'. The 'Start value' is '0'. The 'Increment by' is '1'. Under 'Count type', 'Binary' is selected. Under 'Transitions occur', the 'Count every' field is '250.0' and the unit is 'ns'. There are 'OK' and 'Cancel' buttons at the bottom.

Radix: Binary

Start value: 0

Increment by: 1

Count type

☒ Binary

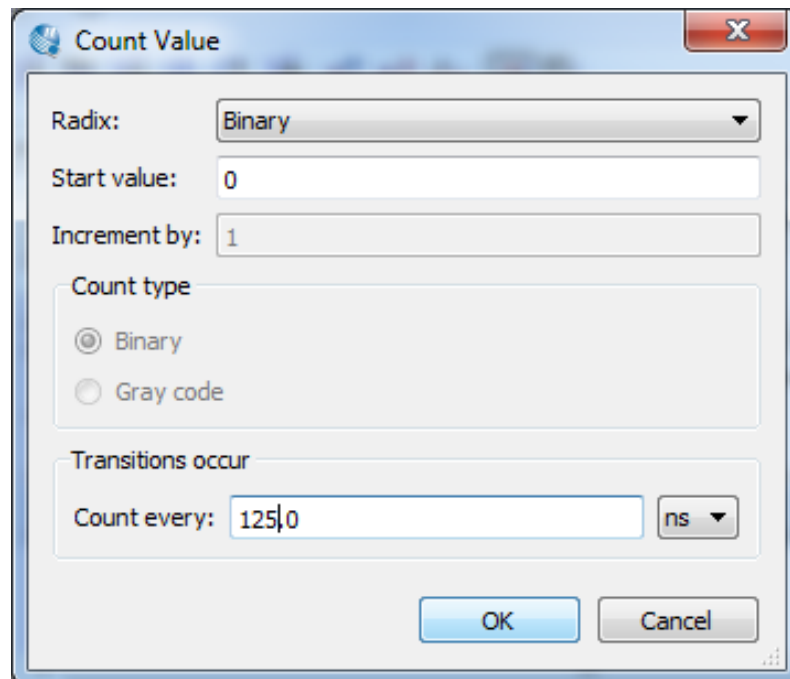
☐ Gray code

Transitions occur

Count every: 250.0 ns

OK Cancel

26) And finally for x2 to count every 125 ns.



The 'Count Value' dialog box for x2 is shown. It has a title bar with a close button. The 'Radix' is set to 'Binary'. The 'Start value' is '0'. The 'Increment by' is '1'. Under 'Count type', 'Binary' is selected. Under 'Transitions occur', the 'Count every' field is '125.0' and the unit is 'ns'. There are 'OK' and 'Cancel' buttons at the bottom.

Radix: Binary

Start value: 0

Increment by: 1

Count type

☒ Binary

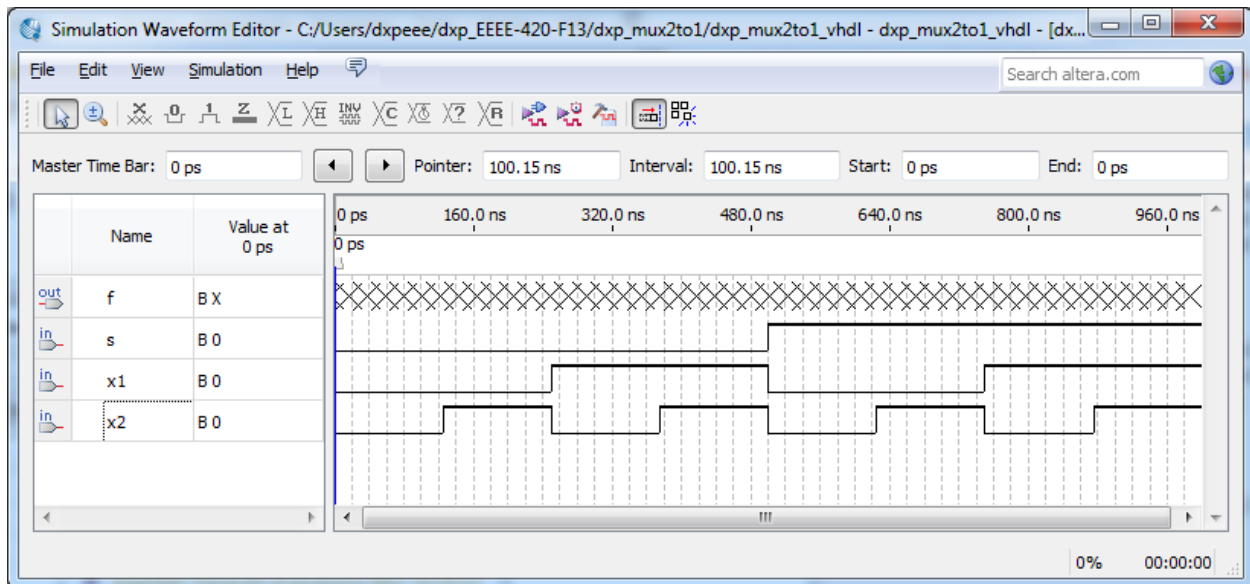
☐ Gray code

Transitions occur

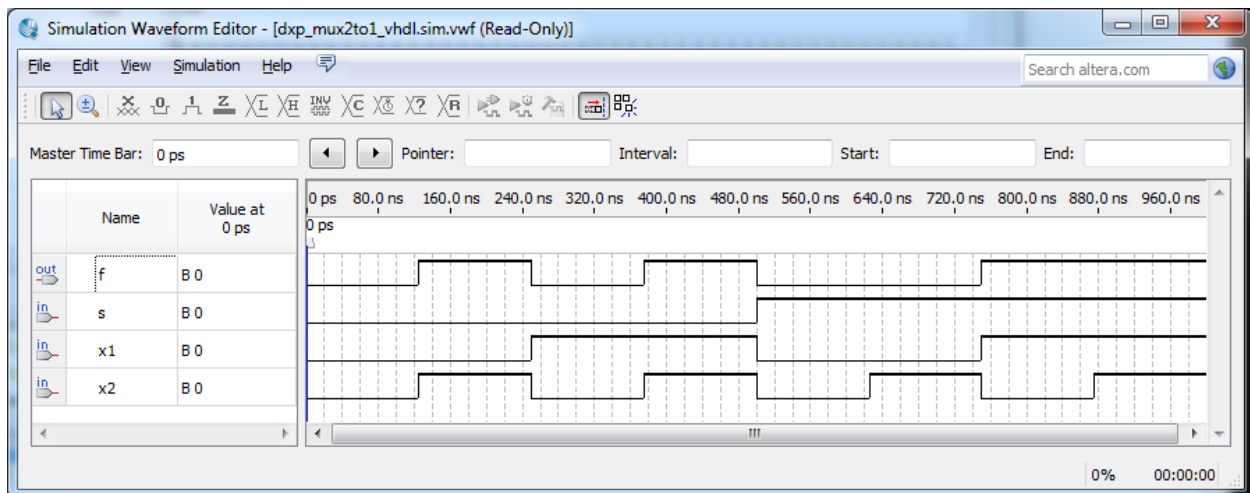
Count every: 125.0 ns

OK Cancel

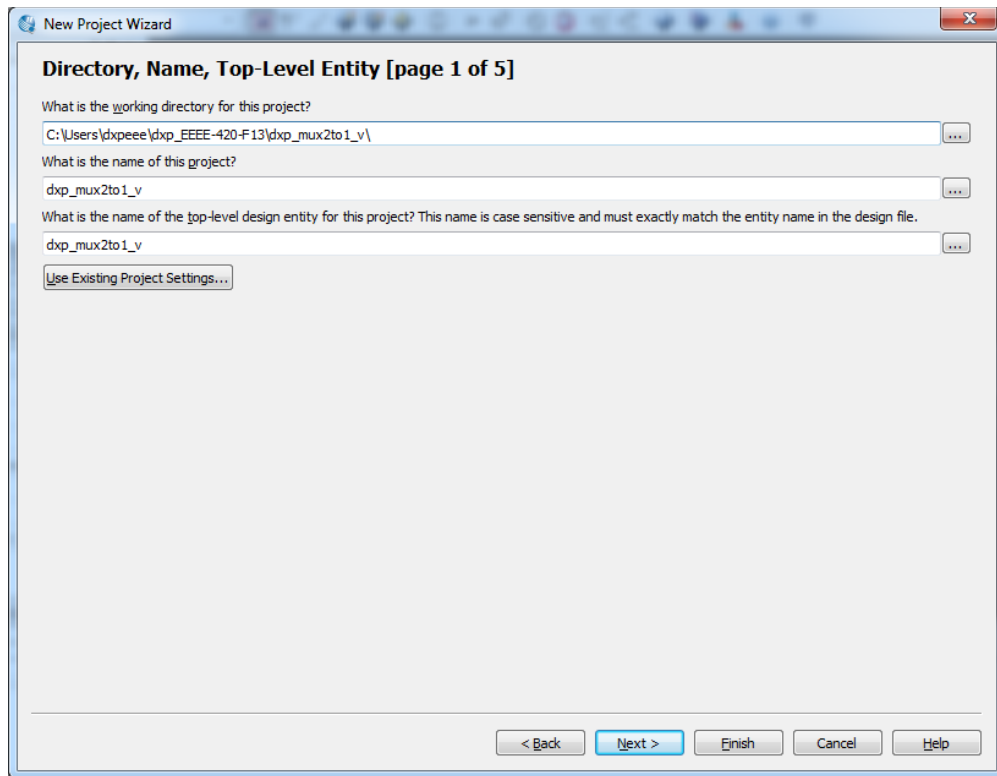
27) Your waveform window should look like the one below now:



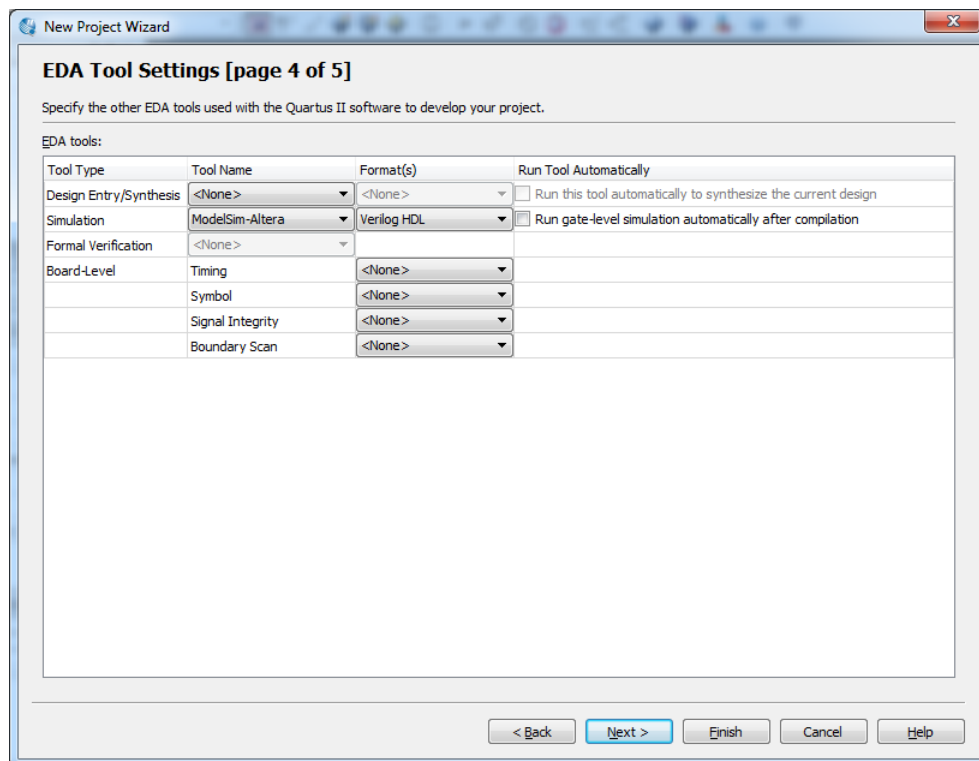
- 28) Observe that by the selection of the frequencies we have made the input stimuli cover all possible eight combinations. At this point click File > Save.
- 29) You are ready to simulate. Choose Simulation > Run Functional Simulation. Remember: functional simulation checks for the logical correctness of your design. It assumes 0 propagation delay through all gates and wires. After the third party ModelSim simulator runs it outputs the result in the form of a Read-Only waveform as below:



- 30) We can now visually verify the correctness of the design. This is a very simple circuit and this is easy to do in this way. However, next time we'll learn a better and more efficient way, i.e. through the use of a testbench.
- 31) Close both waveform windows and the project by going to File > Close Project.
- 32) Next, we will design the same circuit using Verilog. Start by creating a new project with the project wizard and **name the new directory and project fml_mux2to1_v.**

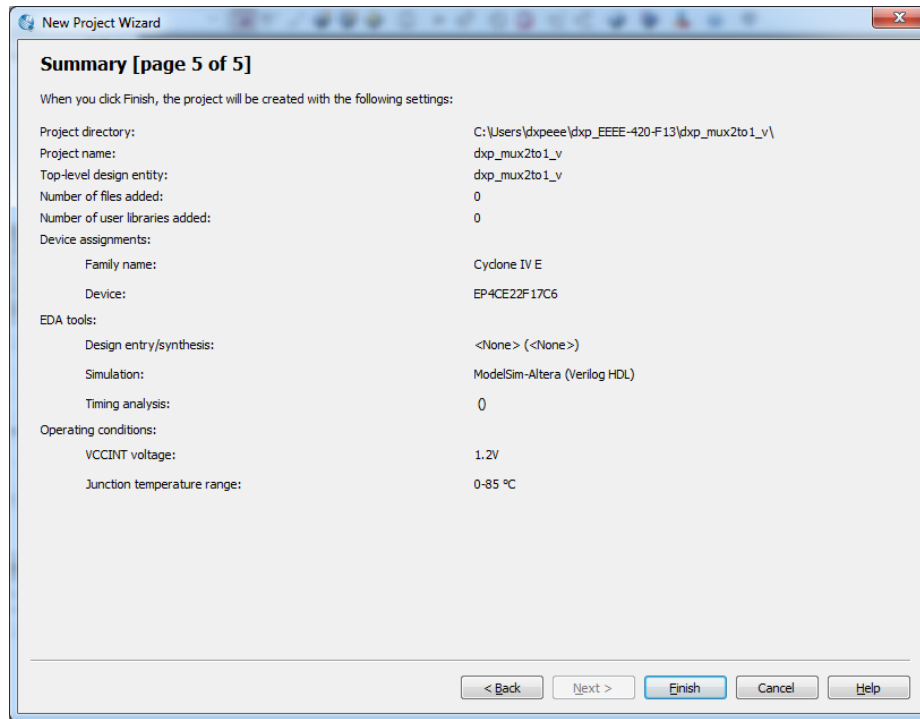


33) Choose the right device and in step 4 of 5 select Verilog HDL for the Format(s).

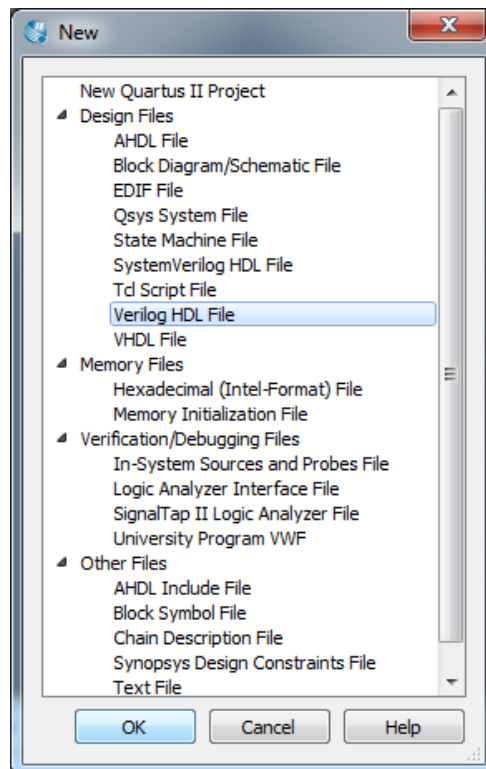


34)

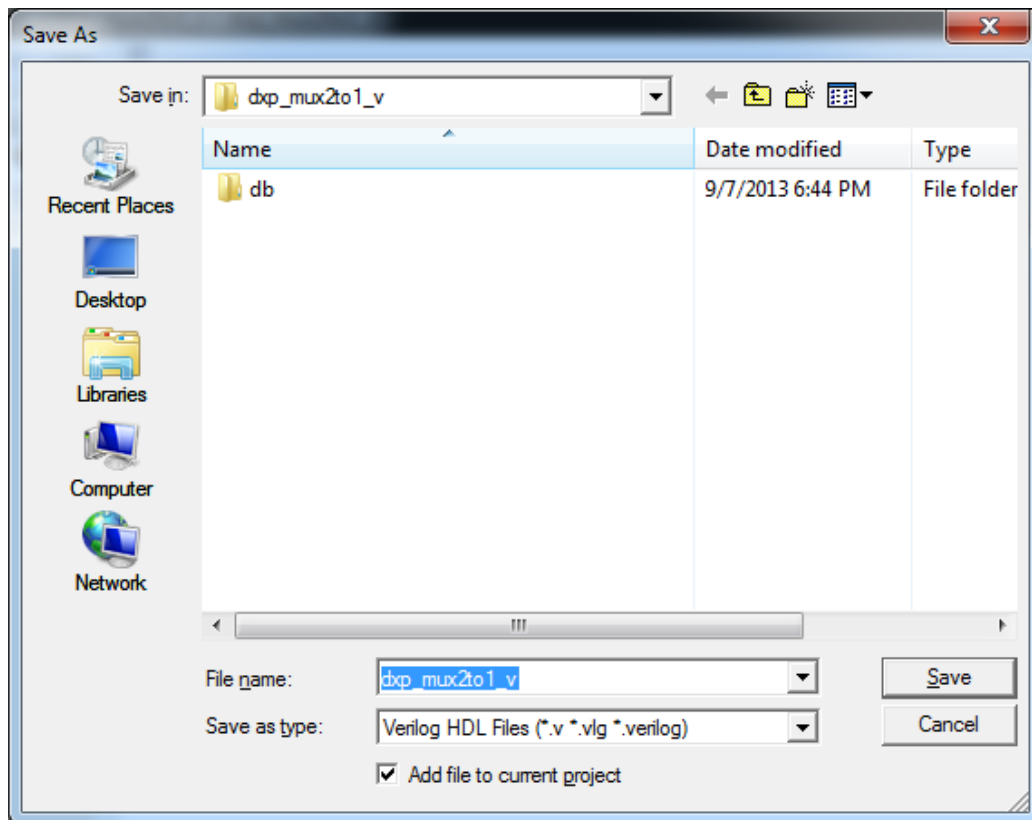
35) Before you click Finish double-check the summary below.



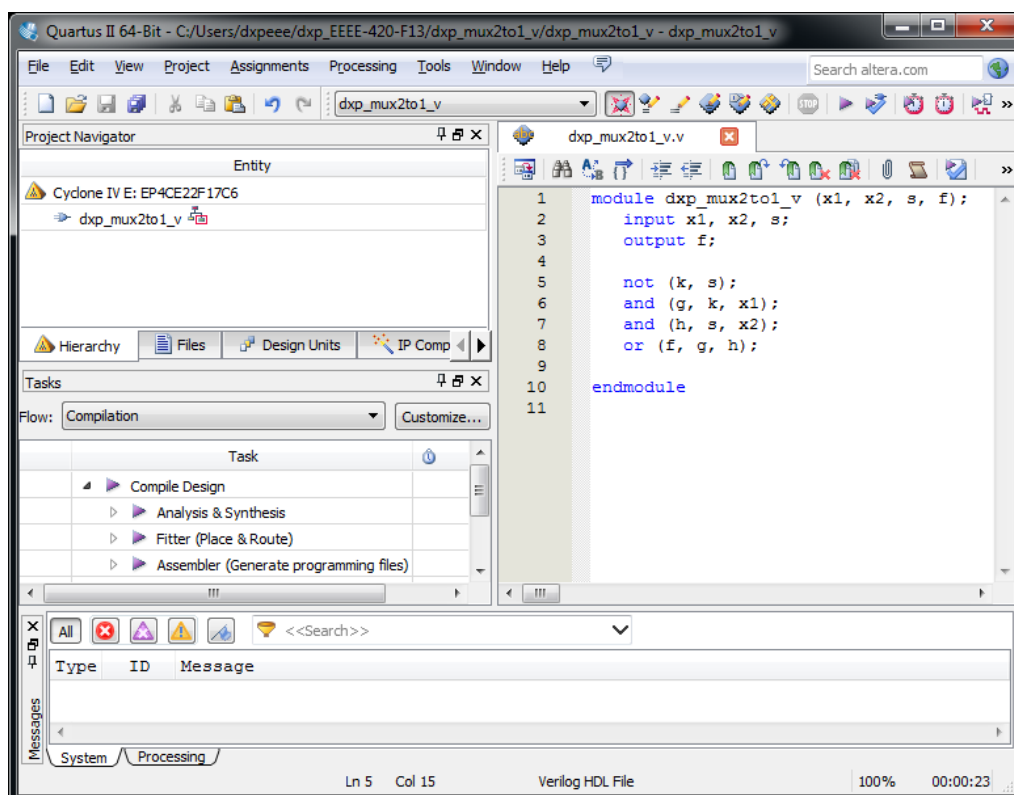
36) Go to File > New > and select Verilog HDL File:



37) Save as:



38) Copy or type the Verilog code below. It has been explained in lecture 6.

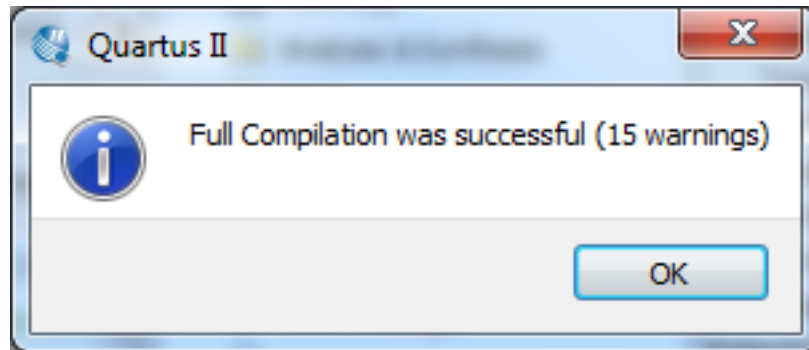


39) If needed, open those lecture notes and spend the time to review the significance of every line of code before you proceed to the next steps. Click Save.

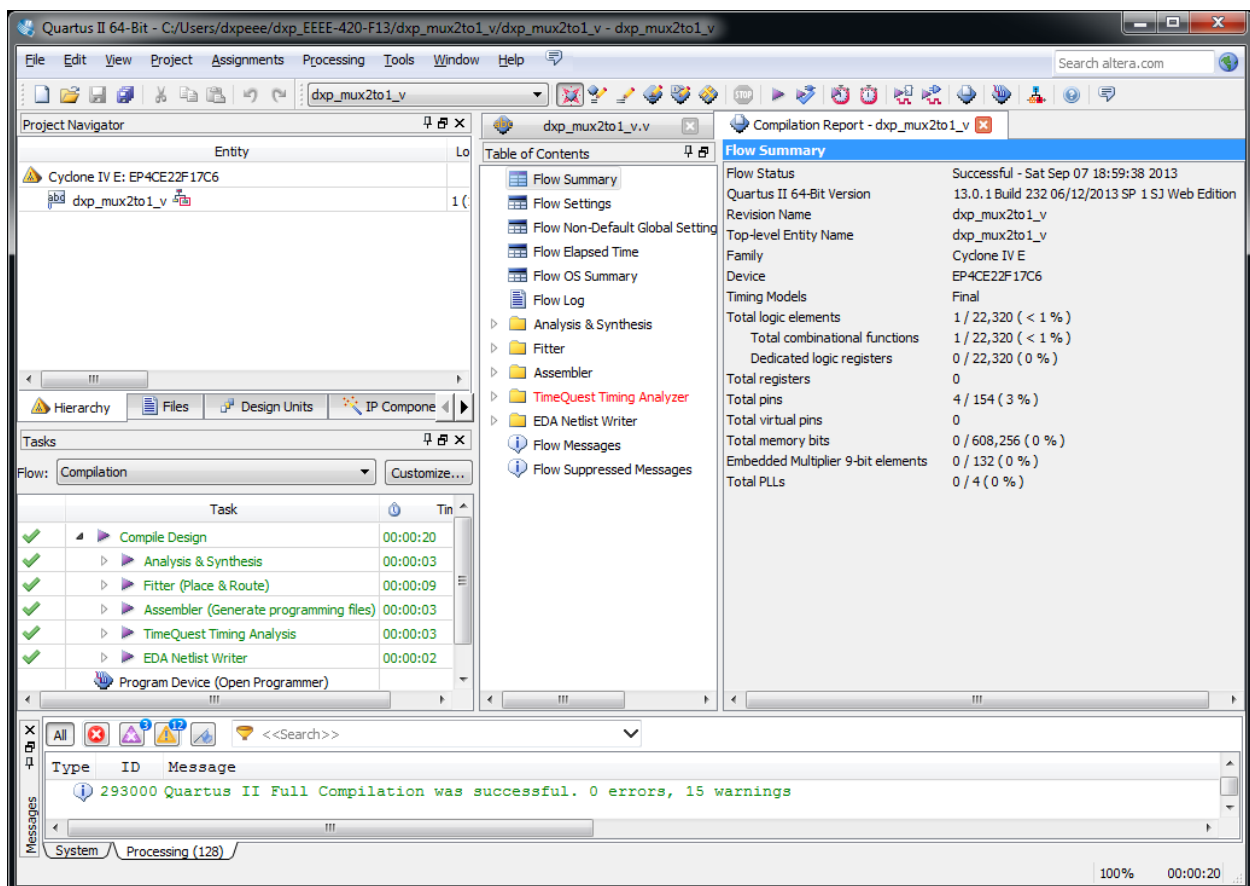
40) Next we want to compile/synthesize the design. You can start this process by doing one of the following:

- Click Ctrl-L
- Click the magenta play button in the upper toolbar
- Or go to Processing > Start Compilation

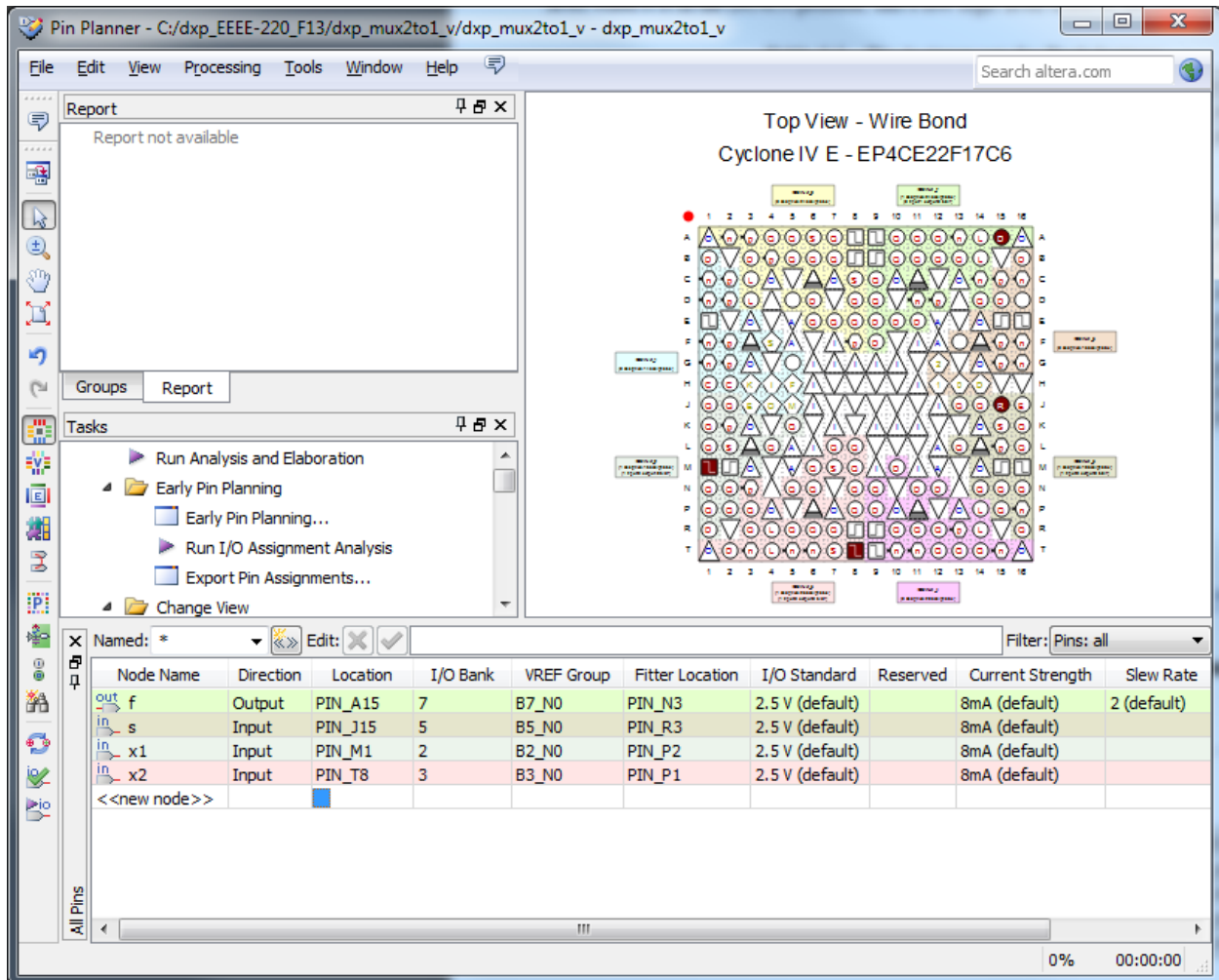
41) If all is going well you will see the window below:



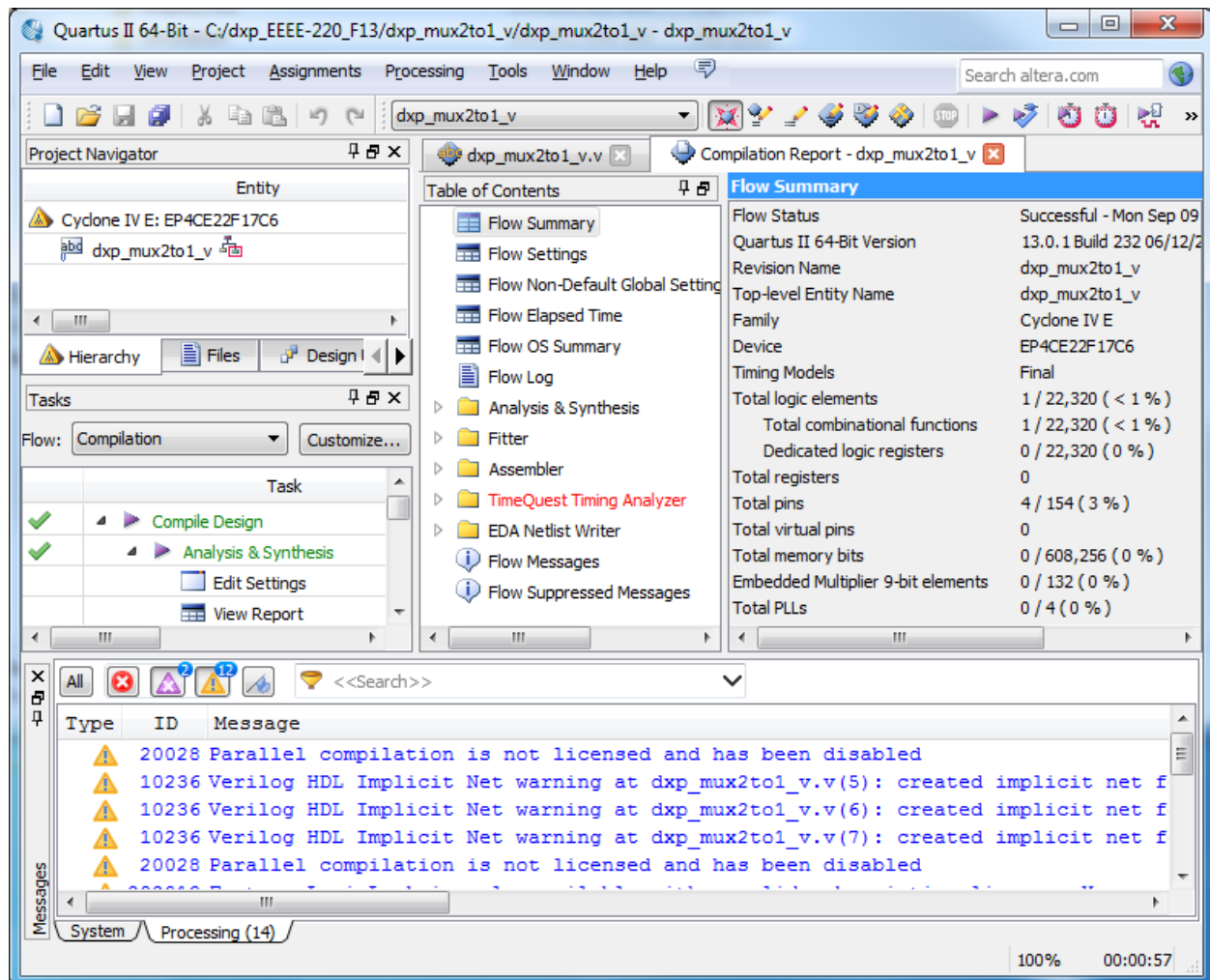
42) Below is the compilation summary:



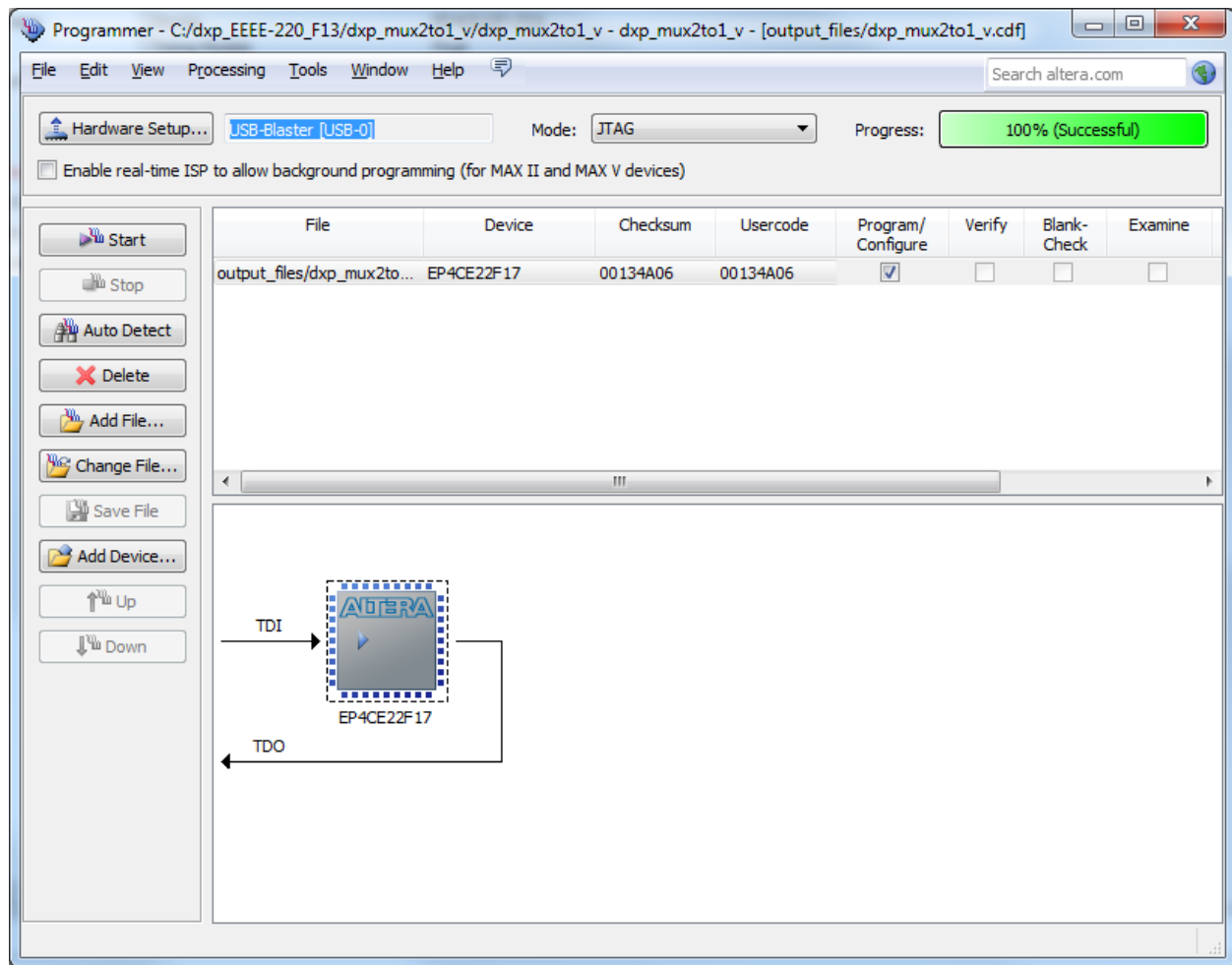
- 43) We are again ready for simulation. At this point you can follow the steps described above between 17 and 30. Make sure you use the name fml_mux2to1_v.vwf for your waveform file!
- 44) We will next test the circuit on the prototyping board. Open Assignments > Pin Planner.
- 45) Select x1 and x2 to be controlled by two DIP switches, s to be controlled by a pushbutton, and f output to an LED. You'll find the pins on pages 13 and 14 of the board user's manual. Once you make the necessary selections, your window should look like the one below.



- 46) Close this window and re-compile again. You may get some warnings, which you can ignore for now. Some refer to the implicitly created internal nodes, others to the fact that we don't have clocks in the circuit because it a pure combinational circuit.



47) Open the programmer from Tools > Programmer, and configure the device.



- 48) Test your circuit on the board by going through all eight combinations of s, x1, and x2.
- 49) Close the programmer window.
- 50) Archive the project by running Project > Archive Project.
- 51) Close this project.
- 52) Open the VHDL project again. Repeat steps 44 to 51 for this project.
- 53) Show your working designs to the TA. Write your report and upload it along with your project archives in the dropbox on mycourses, as described in the lab policy.
- 54) This concludes this week's lab.
- 55) Grading:
 - a. 5 points for VHDL simulation
 - b. 5 points for Verilog simulation
 - c. 5 points for VHDL on board test
 - d. 5 points for Verilog on board test.