

# Quick Tutorial for Quartus II & ModelSim Altera

By Ziqiang Patrick Huang

Hudson 213c

[Ziqiang.huang@duke.edu](mailto:Ziqiang.huang@duke.edu)

## Download & Installation

For Windows or Linux users :

Download Quartus II Web Edition v13.0 (ModelSim Altera included) from <http://dl.altera.com/13.0/?edition=web> , then follow the instruction and install it.

For Mac users:

Either use virtual machines or the computers in the lab (Already installed with all the software you need)

## Create a new project

1. Open Quartus II, select **File > New Project Wizard** to reach the window in Figure 1 and choose the working directory and name for the project. The name of top-level design entity usually matches the project name.

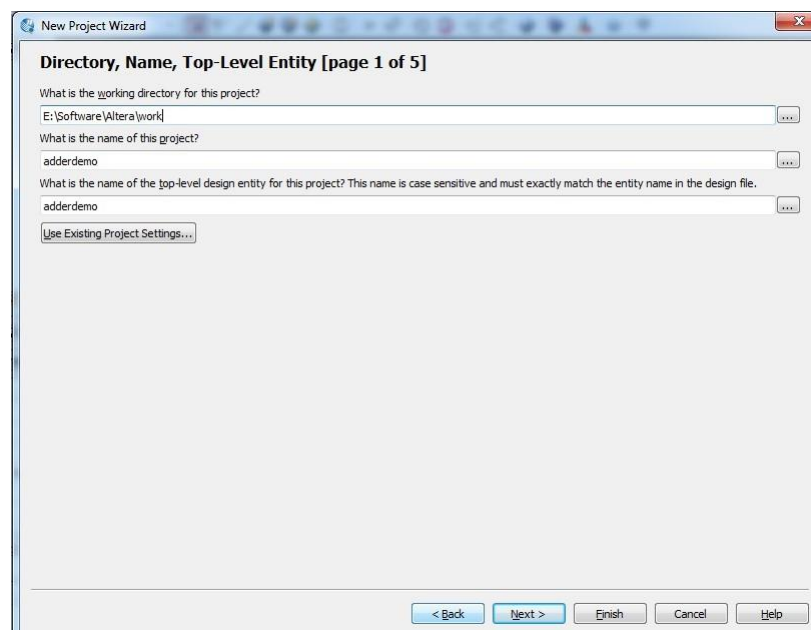


Figure 1

2. Click **Next**, if the working directory does not exist, a pop-up box will appear to ask you to create the directory, click yes and reach the window in Figure 2. Here you could include the existing design files.

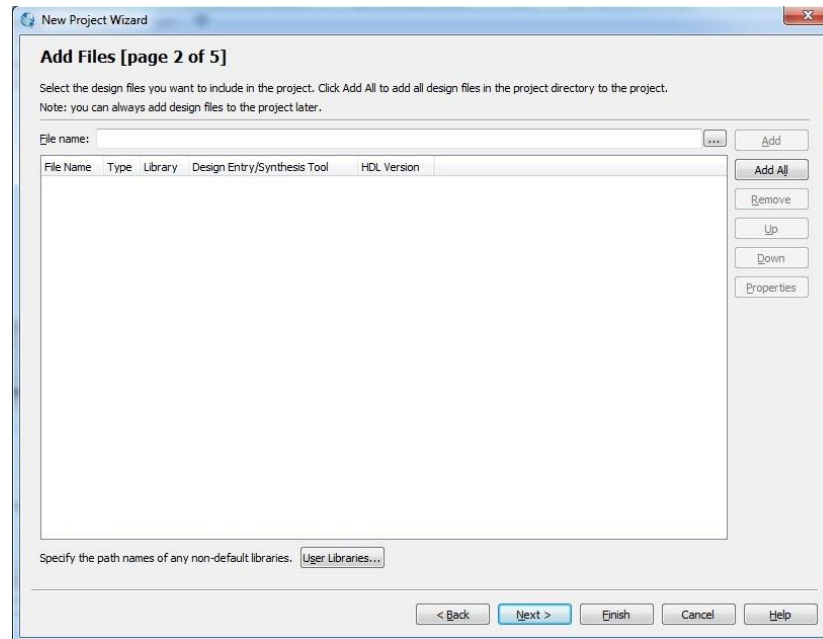


Figure 2

3. After adding all the files you want to include or if you don't have any, click next to reach the window in Figure 3. Here you need to specify the device in which the design circuit will be implemented. Under **"Family"**, select **"Cyclone II"**, under **"Available Device"**, select **"EP2C35F672C6"**, which is the FPGA used on Altera DE2 board in the lab.

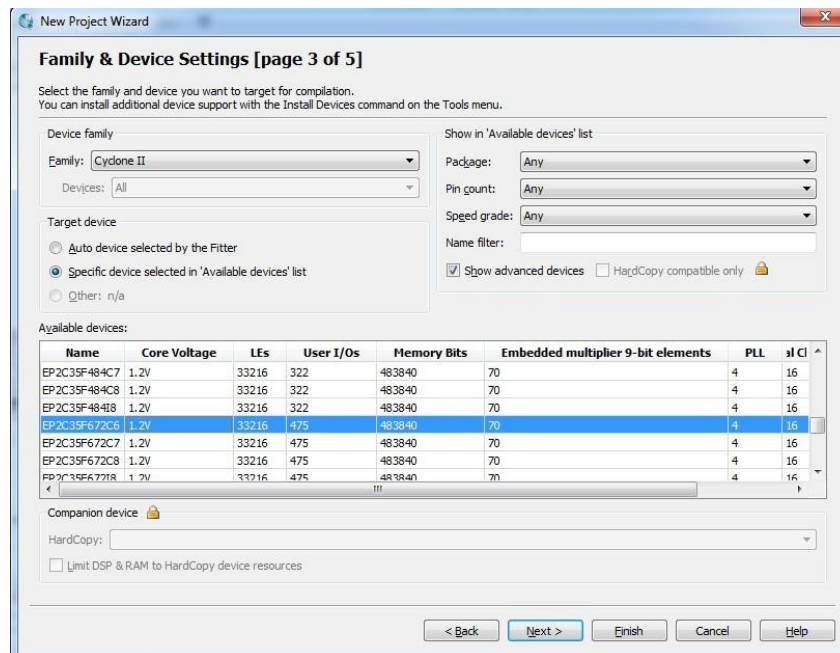


Figure 3

- Click **Next** to reach the window in Figure 4. Here you can specify the EDA tools you want to use. We will use ModelSim in this class, and the format is VHDL, so just leave the default settings

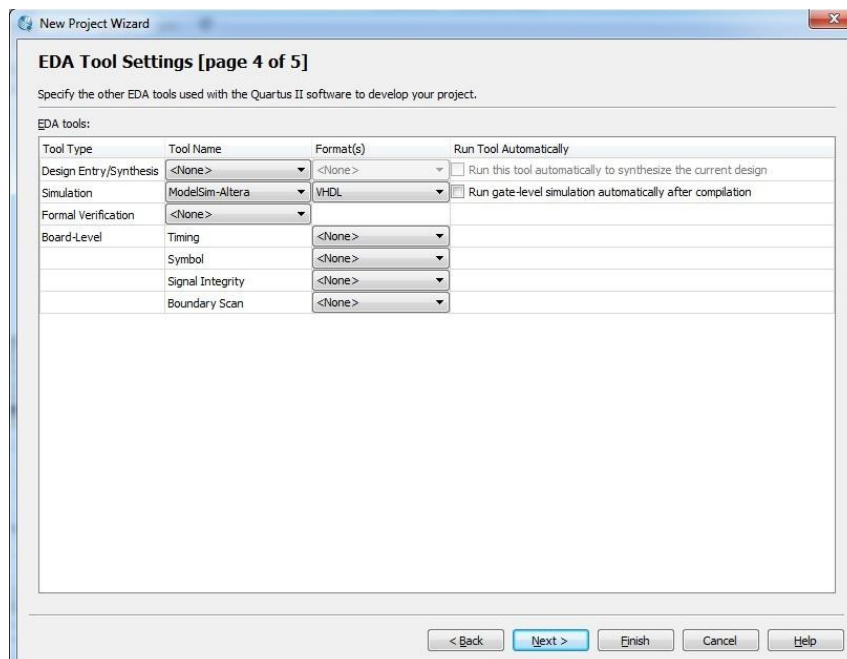


Figure 4

- Click **Next** and a summary of settings will be shown to you as in Figure 5.

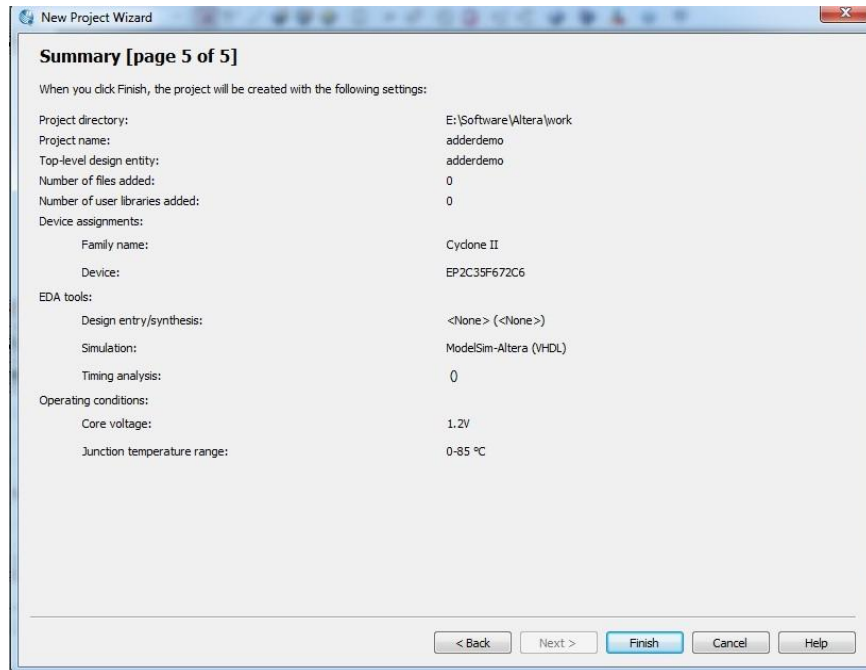


Figure 5

6. Click **Finish** and you will return to the main Quartus II window, under “Project Navigator” sub-window, you can see the project you just created with no design file at this moment, as in Figure 6

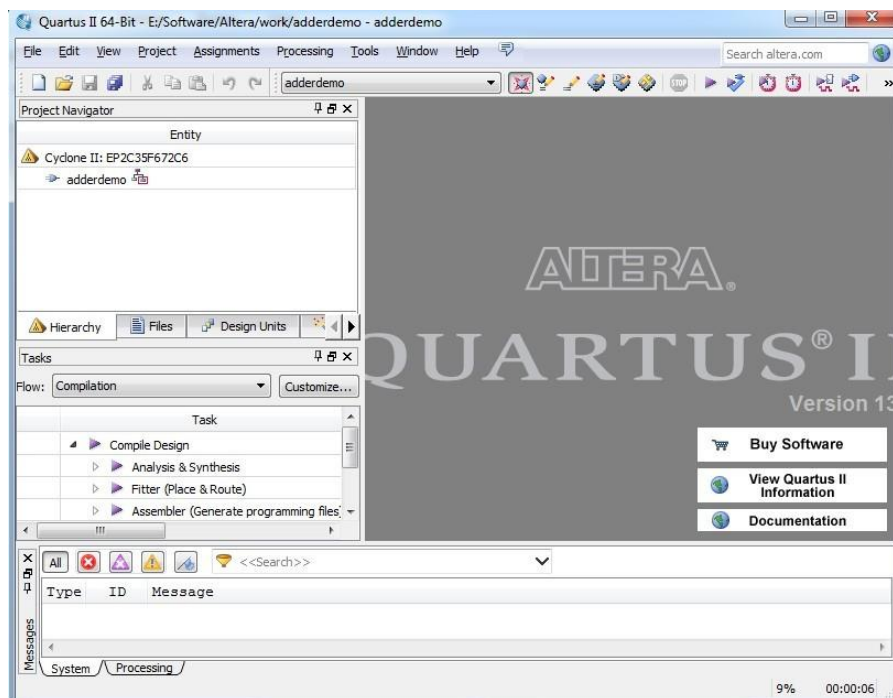


Figure 6

## Write design files in VHDL

### 1. Using the Quartus II Text Editor

Select File > New to get the window in Figure 7, choose VHDL File, and click OK. This opens the Text Editor window. The first step is to specify a name for the file that will be created. Select File > Save As to open the pop-up box depicted in Figure 8. In the box labeled Save as type choose VHDL File. In the box labeled File name type “adderdemo”. Put a checkmark in the box Add file to current project. Click Save, which puts the file into the directory work and leads to the Text Editor window shown in Figure 9, now in the “Project Navigator”, under “Files”, you can see the design file you just created. We’ll use “adderdemo.vhd” file from the course website as an example, you don’t need to understand all the details in the code at this moment.

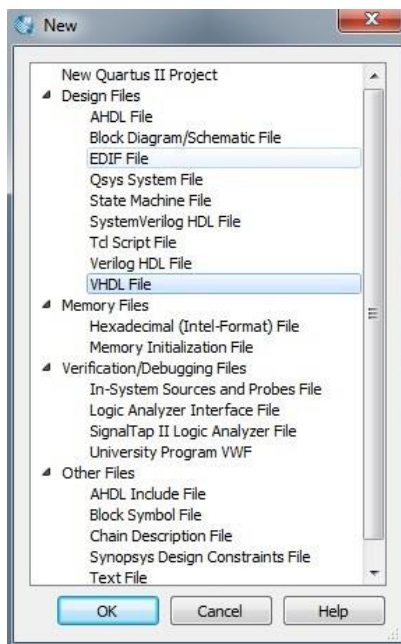


Figure 7

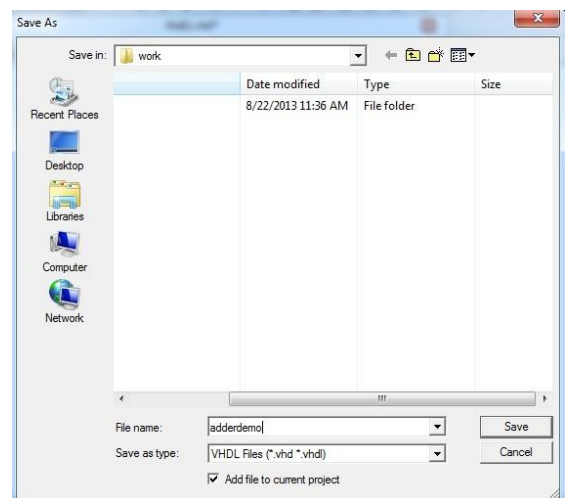


Figure 8

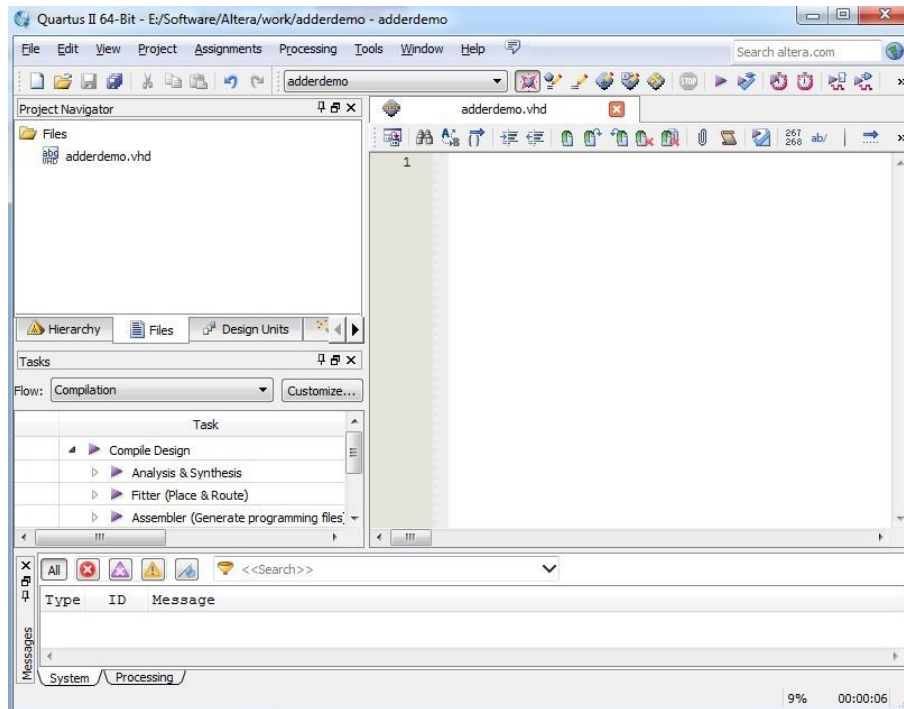


Figure 9

## 2. Using other Text Editor

You can also use any other text editor that stores ASCII files (Emacs, Vim, etc.) to write the source code, but then you need to manually add files into the project. Select **Project > Add/Remove files in project** to reach the window in Figure 10.

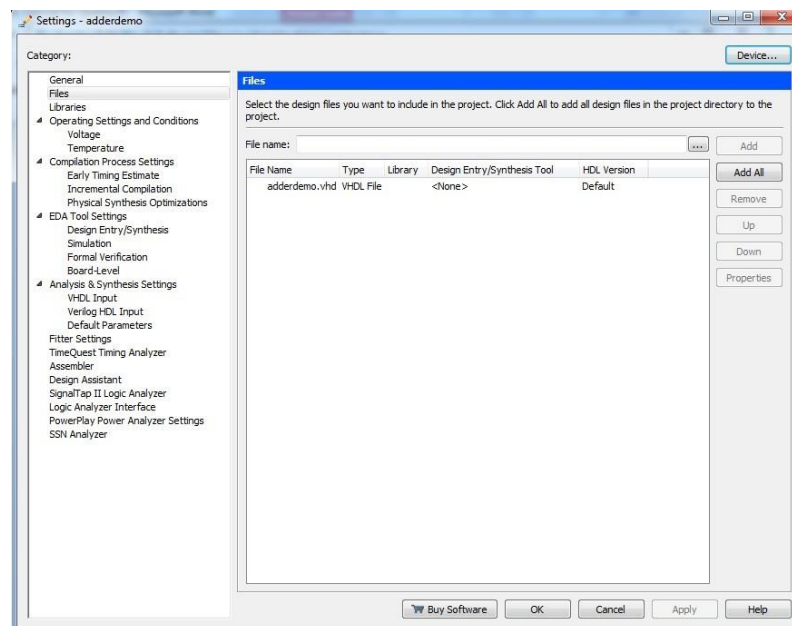




Figure 10

## Compile the designed files

After writing or adding your design files, run the Compiler by selecting **Processing > Start Compilation**, or by clicking on the toolbar icon that looks like a purple triangle . As the compilation moves through various stages, its progress is reported in a window on the left side of the Quartus II display. Successful (or unsuccessful) compilation is indicated in a pop-up box. Acknowledge it by clicking OK, which leads to the Quartus II display in Figure 11. In the message window, at the bottom of the figure, various messages are displayed. In case of errors, there will be appropriate messages given.

When the compilation is finished, a compilation report is produced. A window showing this report is opened automatically, as seen in Figure 11. The window can be resized, maximized, or closed in the normal way, and it can be opened at any time either by selecting **Processing > Compilation Report** or by clicking on the icon . The report includes a number of sections listed on the left side of its window. Figure 11 displays the Compiler Flow Summary section, which indicates that 38 logic elements and 44 pins are needed to implement this circuit on the selected FPGA chip.

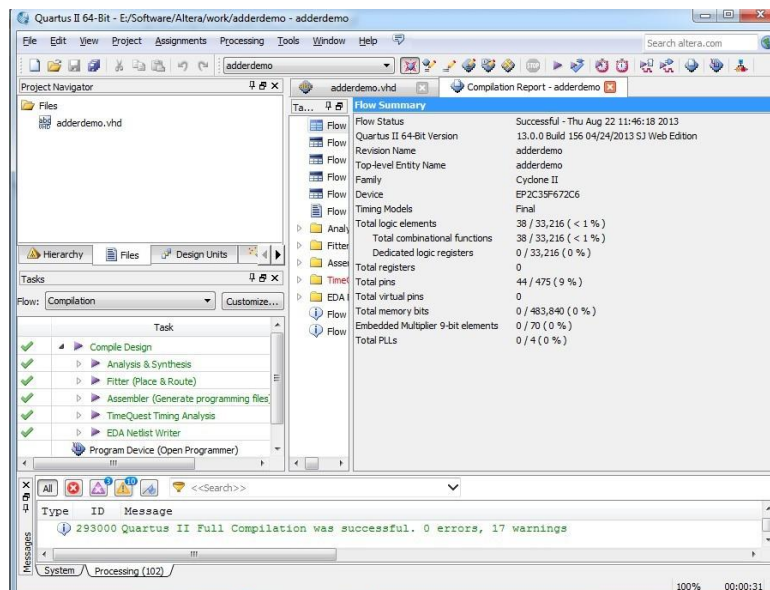


Figure 11

## Simulate with ModelSim-Altera

1. Set up the EDA tool options

Select **Tools > Options** to open up the options window, under **General**, select **EDA Tool Options** on the left as the Figure 12. Click the **Browse** button to the right of

**ModelSim-Altera** in the **EDA Tool** list to specify the location of the ModelSim-Altera executable. Finally click **OK**.

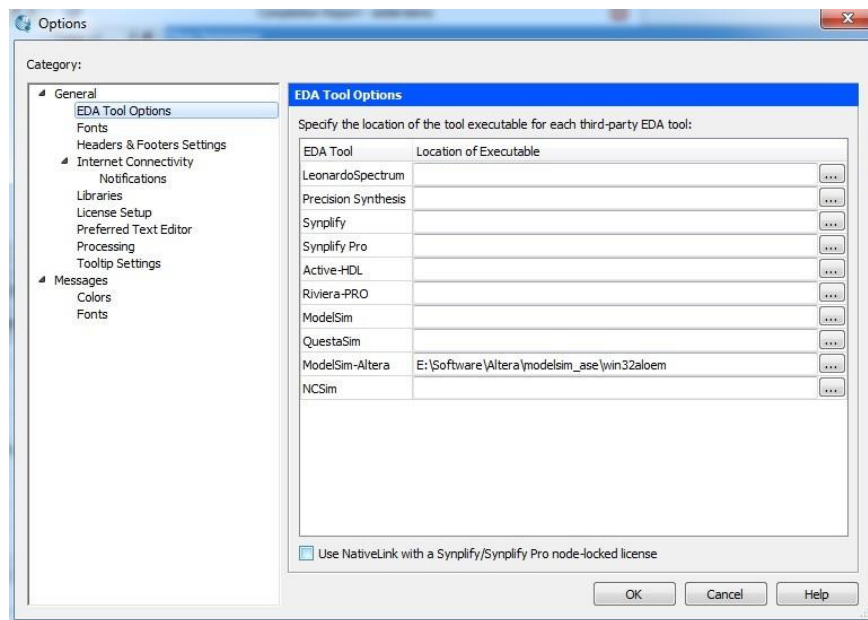


Figure 12

2. Set up the simulation

Select **Assignment > Settings**. In the Settings dialog box, under EDA Tools Settings, select Simulation, as shown in Figure 13. In the **Tool name** list, select **ModelSim-Altera**. Ensure that **Run gate-level simulation automatically after compilation** box is turned off. Under **EDA Netlist Writer settings**, in the **Format for output netlist** list, select **VHDL**. Ensure that the **Map illegal HDL characters**, **Enable glitch filtering** and **Generate Value Change Dump(VCD) file script** boxes are turned off. Under **NativeLink settings**, select **None**. Finally, click **OK**.



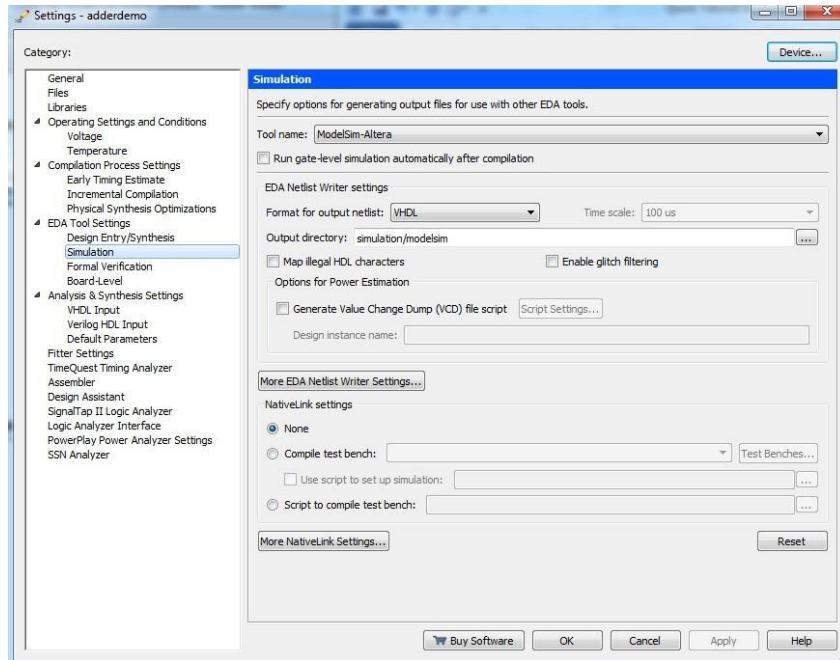


Figure 13

3. Running ModelSim-Altera from Quartus II  
**Select Tools > Run Simulation Tool > RTL simulation** to start ModelSim within Quartus II, as shown in Figure 14.

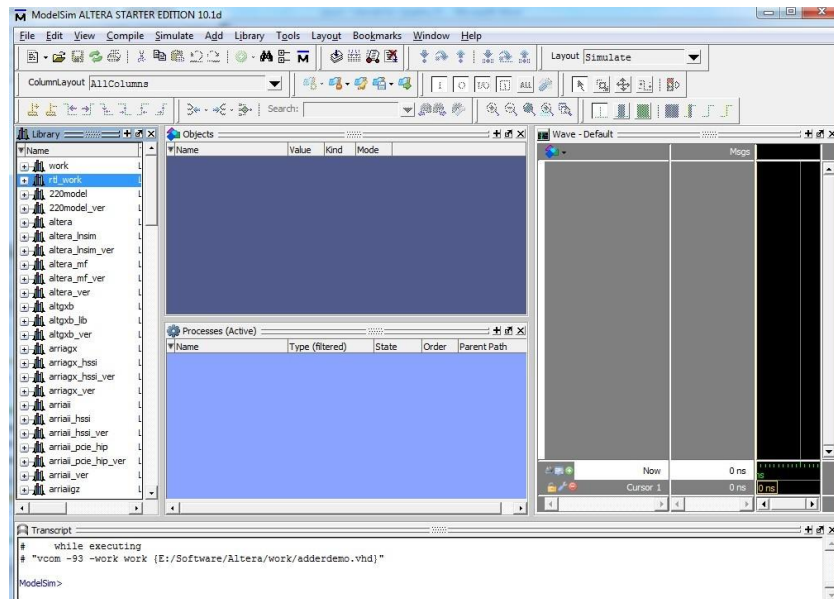


Figure 14

4. Compile  
 Select **Compile > Compile**, in the pop-out window shown in Figure 15, choose adderdemo.vho, click **Compile**, then click **done**.

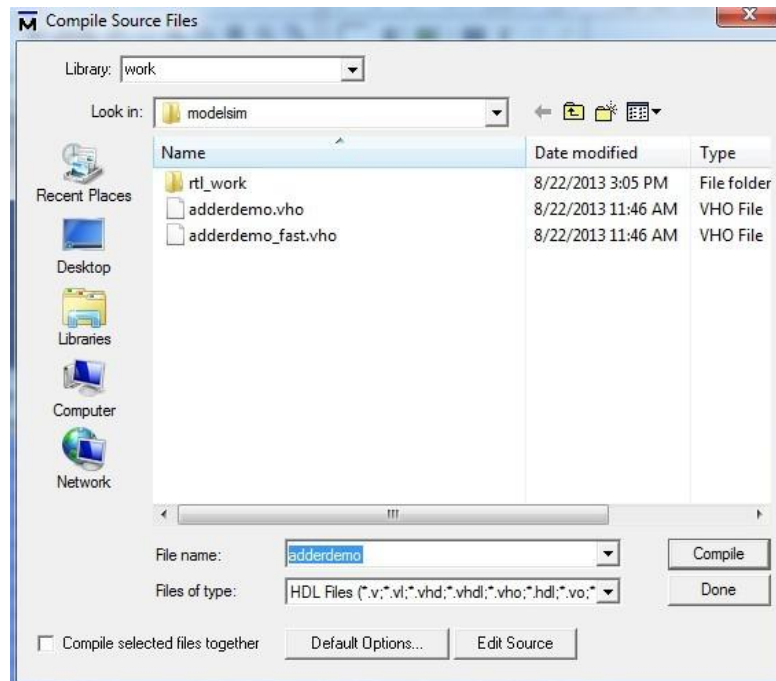
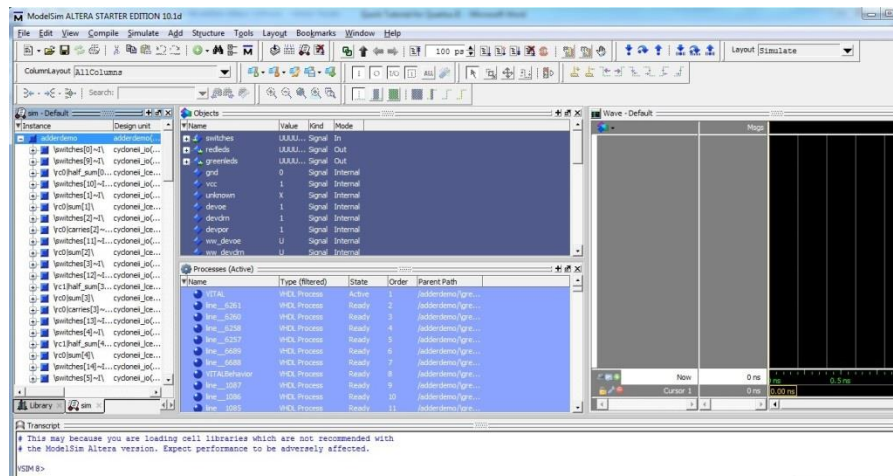


Figure 15


5. Initialize the simulation using command line function.

On the **transcript** window at the bottom, type command: `vsim work.adderdemo`

The format is “**vsim <library\_name>.<design\_unit>**”, then you will see the sim tab and the object window initialized with all the signals in your design, as show in Figure 16.





After forcing all the input signals, the design is ready for simulation. Find the button labeled **Run** . Clicking this will run the simulation for the default time of 100ns. Clicking the Run button again will run the simulation from the previous stopping point, NOT from time 0. You can test a different set of inputs by switching to the signals window, applying new forces, and running the simulation again. You can verify the result in the wave form as shown in Figure 19.

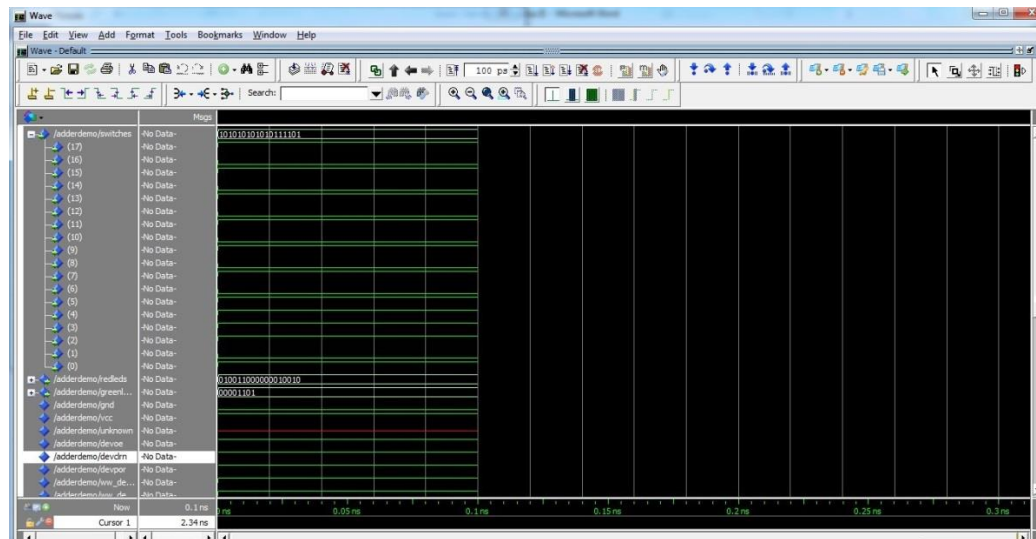


Figure 19

9. Use command line functions and write “Do” script to simulate

When simulating large designs, it is often too cumbersome to manually force a signal using the graphic interface each time you want it to change. ModelSim remedies this by providing a long list of simulator commands. The complete list can be found in the ModelSim help documentation, but for purposes of this tutorial we will cover some of the most commonly used commands.

First do **restart –force –nowave** to restart simulation from time 0.

Then do **add wave \*** to add all the signals from the current simulation. You can add –radix/hex to change how it shows up(hex or decimal).

Then use **force –deposit <signal>** to force the initial state of the signal, for example, **Force –deposit switches 16#0** set the switches signal to 0, Note that 16#0 means 0 as a hex number. In this simple example, we don’t have to set clock, but in most of the designs you will have to do it. For example if you want to set the clk signal to 20ns per cycle in the top-design unit named cache, you do **force –deposit /cache/clk 1 0, 0 {10 ns} –repeat 20 ns**, note “/” is used to show hierarchy.

Your design is also likely to have a reset signal and you should set it at the beginning of your simulation, you do

```
force -deposit /cache/rst 1      -- set the reset signal to 1  
run 20 ns                        -- run for one cycle to let it take effect  
force -deposit /cache/rst 0      -- set it back to 0
```

Then you can start change the inputs and time to process them

```
force -deposit switches 16#235  
run 40 ns  
etc ...
```

To see the result of the simulation, do

```
view wave
```

You can write comments with # ..... #  
**#Initialize state#**

ModelSim allows you to automate sequences of commands using DO files. The files are simple in structure, containing only comments starting with a # and a sequential list of commands. To create a new DO file, navigate to the ModelSim main window and choose **File > New > Source > Do**. This will open the editor window with a blank document. Type the following in the editor and then save the file as "tutorial.do". To execute the file, simply type command  
**do tutorial.do**

## Pin Assignment

If the simulation is successful, we can go back to Quartus II to continue the rest of the work. During the previous compilation, the Quartus II Compiler was free to choose any pins on the selected FPGA to serve as inputs and outputs. However, the DE2 board has hardwired connections between the FPGA pins and the other components on the board. We will use the 18 toggle switches labeled from SW<sub>0</sub> to SW<sub>17</sub> on the left bottom of the board, to provide the external inputs, and the red and green led lights above the switches to provide outputs.

Pin assignments are made by using the **Pin Planner**. Select Assignment > Pin Planner to reach the window in Figure 20.

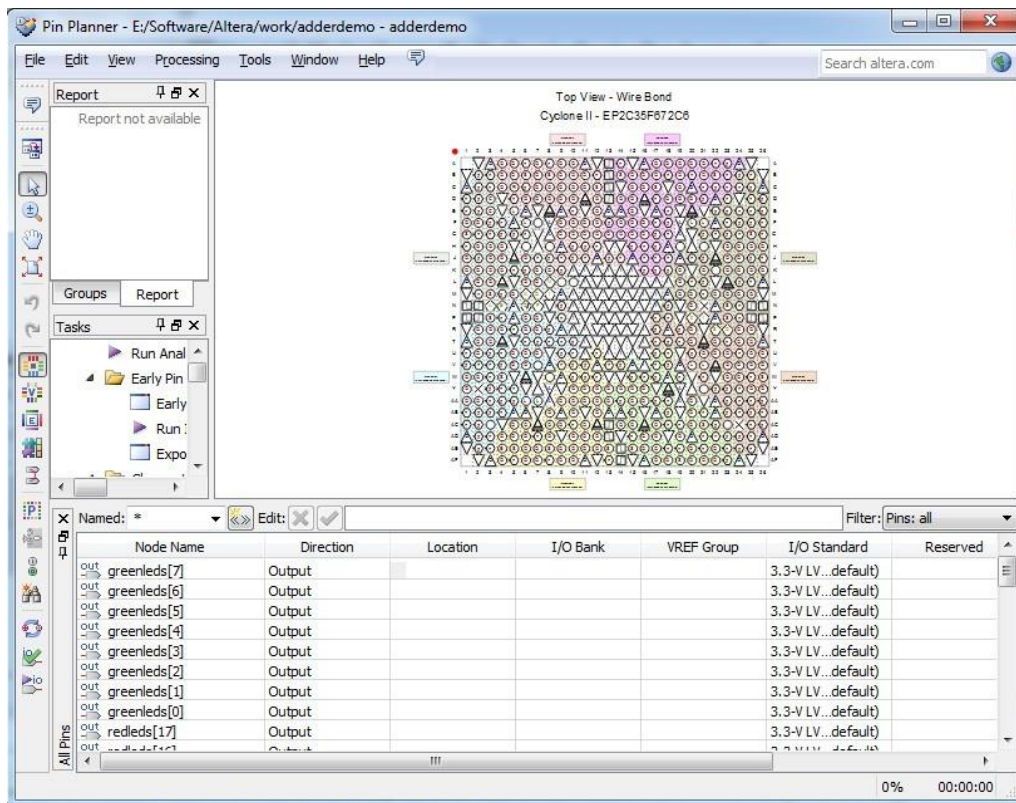


Figure 20

On the bottom you can see all the inputs and outputs are listed. Under each “**Location**”, type in the corresponding pin number for the FPGA. (You can find a detailed Pin table for the board at [http://www.terasic.com.tw/attachment/archive/30/DE2\\_Pin\\_Table.pdf](http://www.terasic.com.tw/attachment/archive/30/DE2_Pin_Table.pdf). As shown in figure 21.



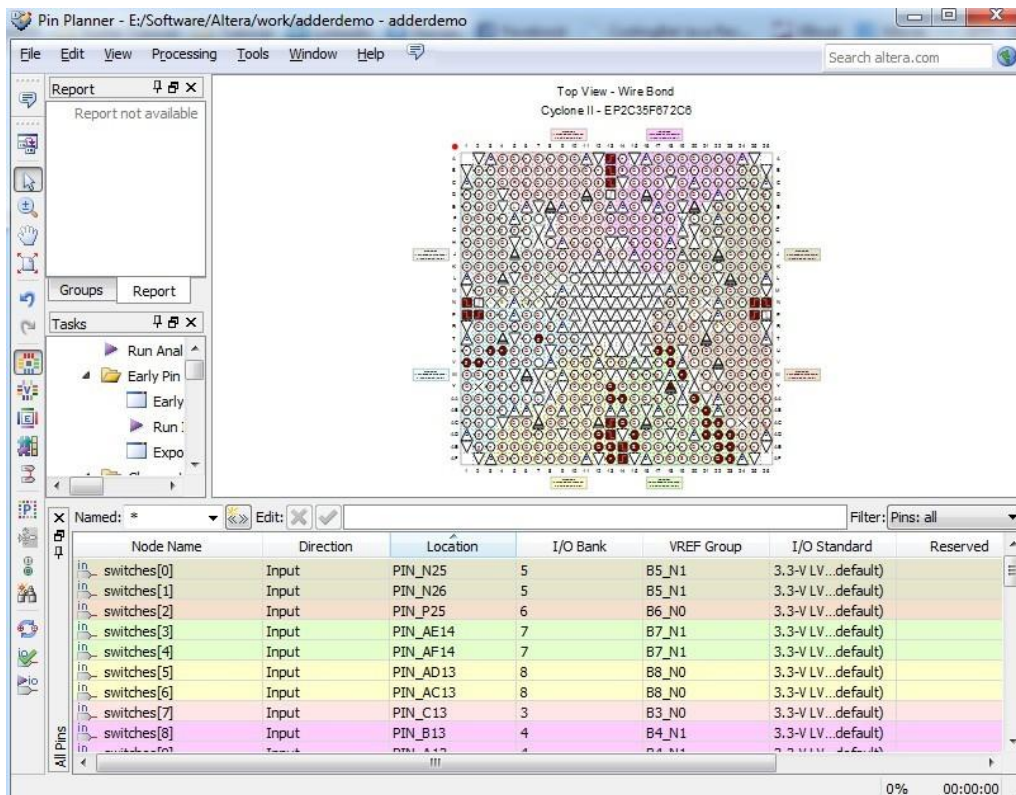


Figure 21

## Programming and configure the FPGA device

The FPGA device must be programmed and configured to implement the designed circuit. The required configuration file is generated by the Quartus II Compiler's Assembler module. Altera's DE2 board allows the configuration to be done in two different ways, known as JTAG and AS modes. The configuration data is transferred from the host computer (which runs the Quartus II software) to the board by means of a cable that connects a USB port on the host computer to the leftmost USB connector on the board. To use this connection, it is necessary to have the USB-Blaster driver installed.

Select **Tool > Programmer** to reach the window in Figure 22. Here it is necessary to specify the programming hardware and the mode that should be used. If not already chosen by default, select JTAG in the Mode box. Also, if the USB-Blaster is not chosen by default, press the Hardware Setup... button and select the USB-Blaster in the window that pops up, as shown in Figure 22.

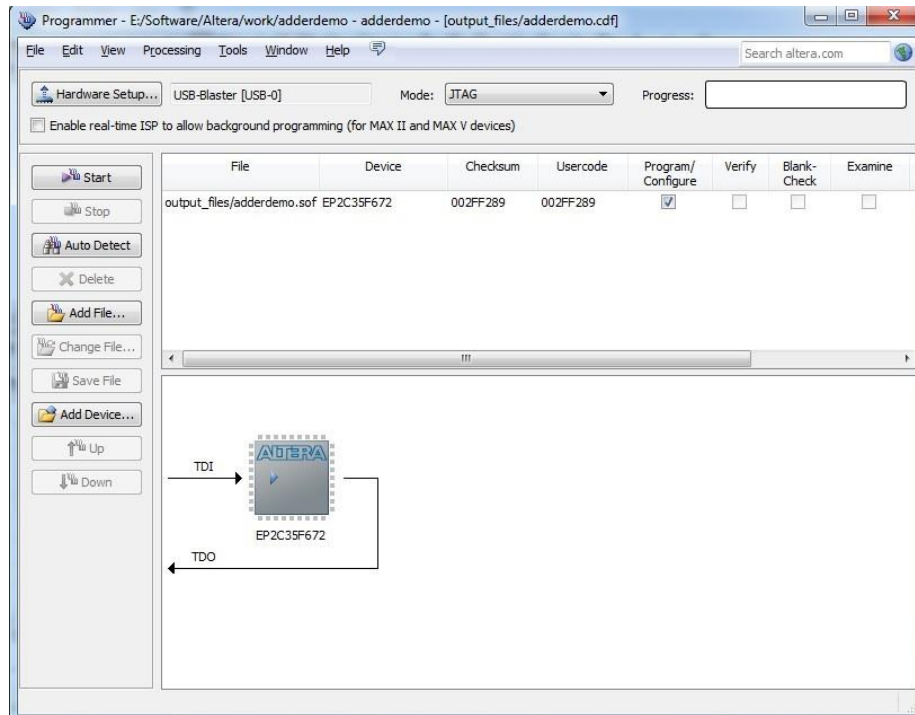


Figure 22

Finally click start to start programming the FPGA, after it finished, you will see the progress showing “100% successful” on the upper right corner.

Then you can use the Altera DE2 board to test your design !

## Acknowledgement

This tutorial is for Course ECE 550D: Fundamentals and Computer Systems Engineering. The material is derived from Altera User Guide: “Getting started with Quartus II simulation using the ModelSim-Altera” and “Quartus II Simulation with VHDL Designs”.