ALTERA INSTALLATION

CSE 141L – UC San Diego

January 22, 2013

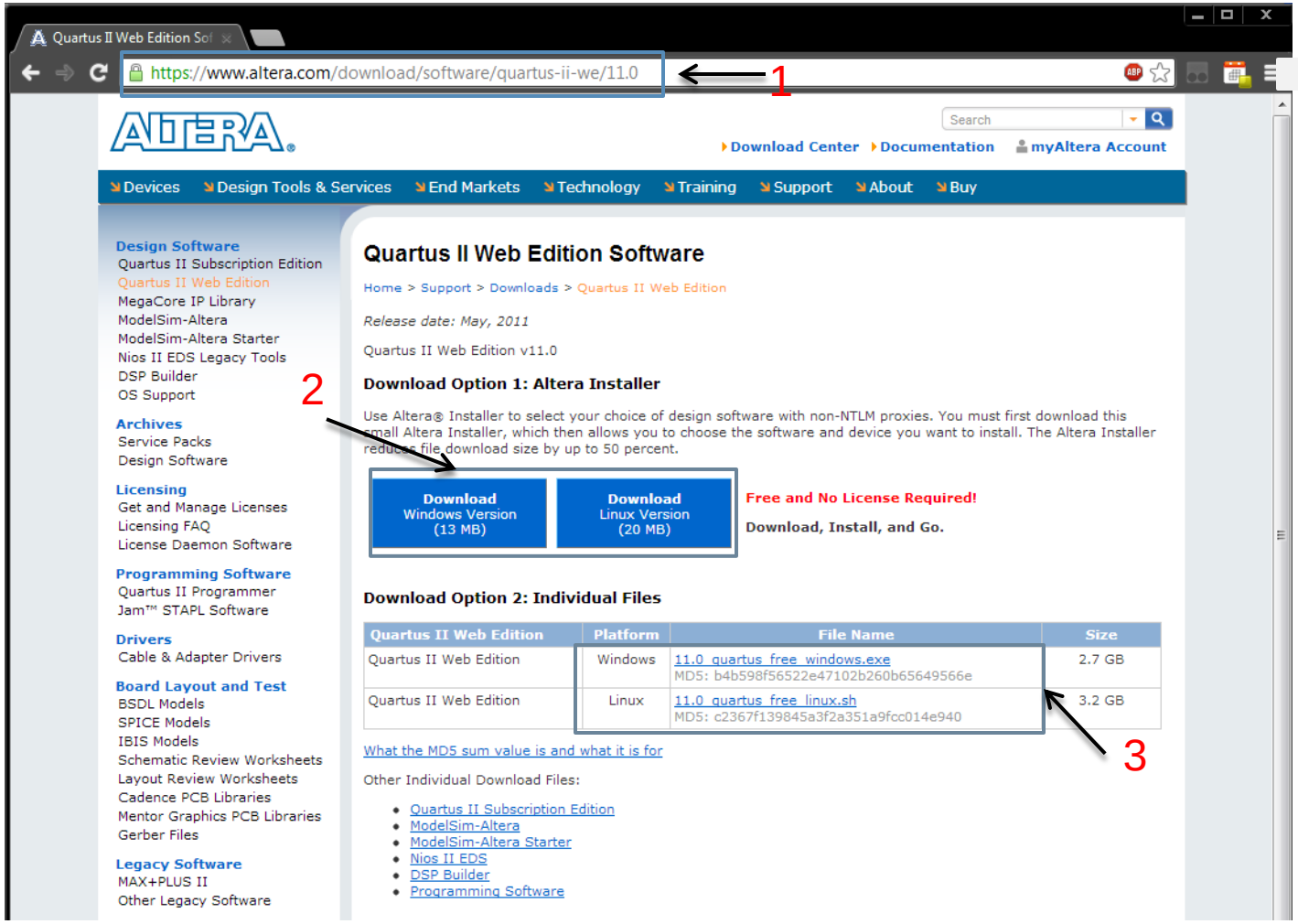
Introduction

- Fortunately most of you have already used Altera in CSE 140L, so things will largely be the same for you.
- If you haven't, its basically an IDE for hardware design
- At least B250 and B260 have Altera installed, but the labs are very crowded this quarter so you may want to install it yourself as well.

Altera Installation

- Step 1: Go to https://www.altera.com/download/software/quartus-ii-we/11.0
- Step 2: Select whatever version you choose (We will support the Windows version since that is what is on the machines in the lab, but there are Linux installs available as well.
- Step 3: Option 1 requires Java, if this is not to your liking (I wouldn't blame you given recent events), you can download the file directly.

Altera Installation



Altera Installation

- Step 4: When asked to create an account, you can just choose the “One-Time Access” option, using an email and “UCSD” for the business.
- Step 5: IF you did step 3 and downloaded the file directly, you may also need to download Modelsim separately. You can find Modelsim here: https://www.altera.com/download/software/modelsim-starter/11.0
  - Download the stock 11.0 version first, use sp1 only if there are problems.

Altera Installation

- Step 6: Once the process to create your account is done, you can sign in to install the file.
- Step 7: Downloading a 2.7 GB takes time…try downloading on campus. Grab a coffee.
  - Kidding aside, this highlights another issue with Altera, installation takes time, do it asap.
- Step 8: If you used the download manager you should have Altera auto-install for Quartus II **Web Edition** and Modelsim-Altera **Starter Edition**
  - There are paid versions of this, make sure to get the free ones.

Altera Installation

- Step 8b: If you downloaded manually make sure you have these two programs and ensure that the Cyclone and Stratix FPGA libraries are installed as well.
- Step 9: Once you begin the installation…you can safely step away.
  - You can probably cook a nice dinner, this can take well more than an hour to install fully.
  - As such this is another reminder to install this ASAP, the basement labs appear swamped this quarter so space is at a premium.
- Step 10: You do not need a license, so if something pops up asking for one ignore it.
- Step 11: Fin!

ALTERA/VERILOG TUTORIAL

CSE 141L – UC San Diego

January 22, 2013

Altera Tutorial

- This tutorial is adapted from the CSE 141L Lab 1 from Summer 2012 (available here: http://cseweb.ucsd.edu/classes/su12/cse141L-a/lab1.html).
- It will show how to set up a project, write a couple Verilog files, then test them.
- It is NOT designed to show you how to do your lab, as such the ALU is very simple, but it may give you ideas on how to implement your project.

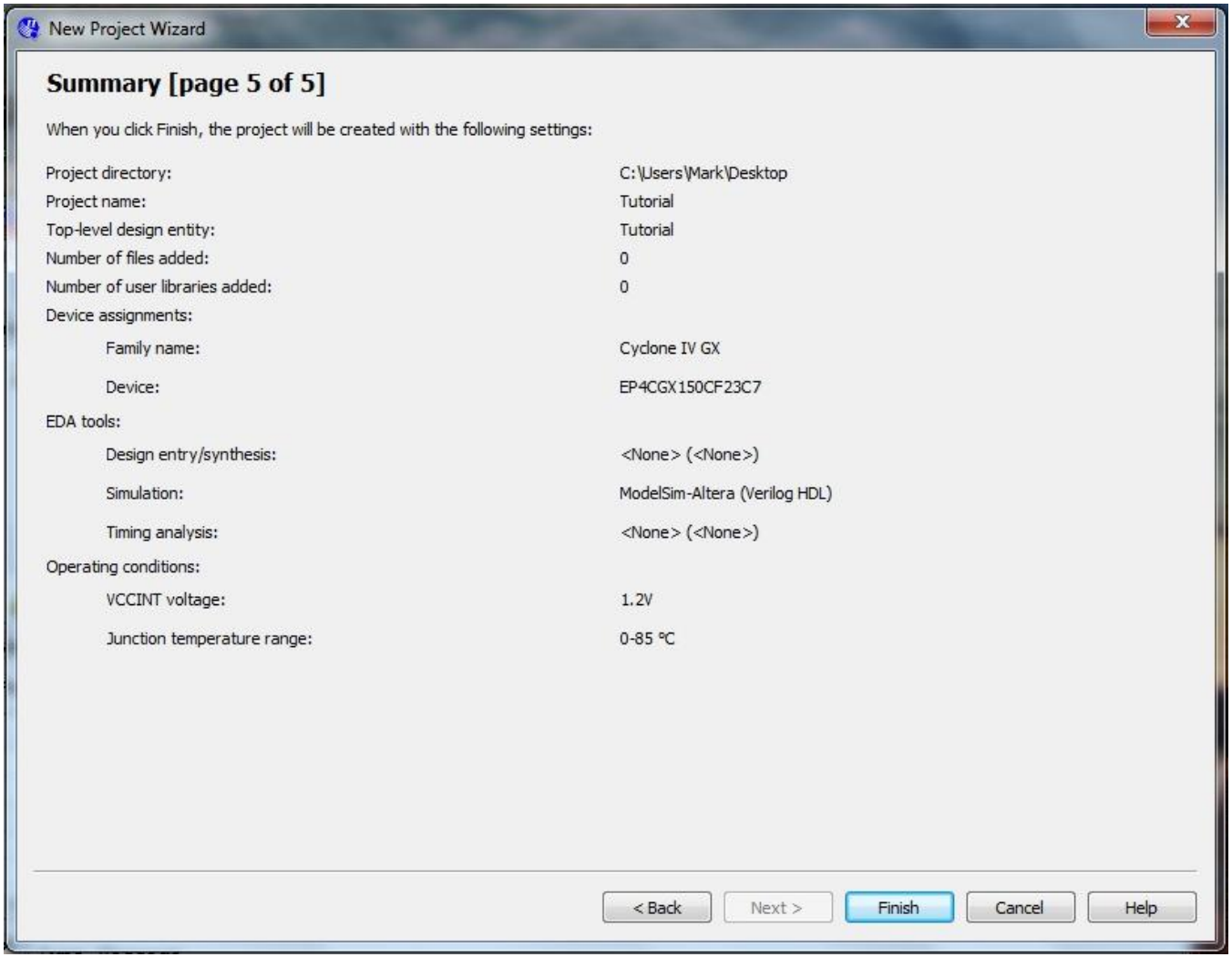
Altera Tutorial

- Some additional (and more detailed) tutorials can be found on Piazza.
  - One covers how to set up a project in Altera
  - The other covers how to do simulations using ModelSim
- Even further help can be found here: https://sites.google.com/a/eng.ucsd.edu/using-the-altera-tools/

Altera Tutorial

- Step 1: With Altera open, go to File -> New Project Wizard…
- Step 2: Type the name of the project and the directory you want to place it in. Click next to pass step 2.
- Step 3: When you reach Step 3: Select the Cyclone II family and the EP2C35F672C6 chip.
- Step 4: For “Simulation”: Click the tool name drop box and select “Modelsim-Altera”, for the format: “Verilog HDL”
- Step 5: Verify that you did the above on the Summary screen.

Altera Tutorial



Altera Tutorial

- Sidebar: What the is a Cyclone II EP2C35F672C6?
  - Short answer: It is an FPGA designed by Altera that allows you to translate your schematic/verilog onto an actual chip.

Table 1. Cyclone IV GX FPGA Family Overview

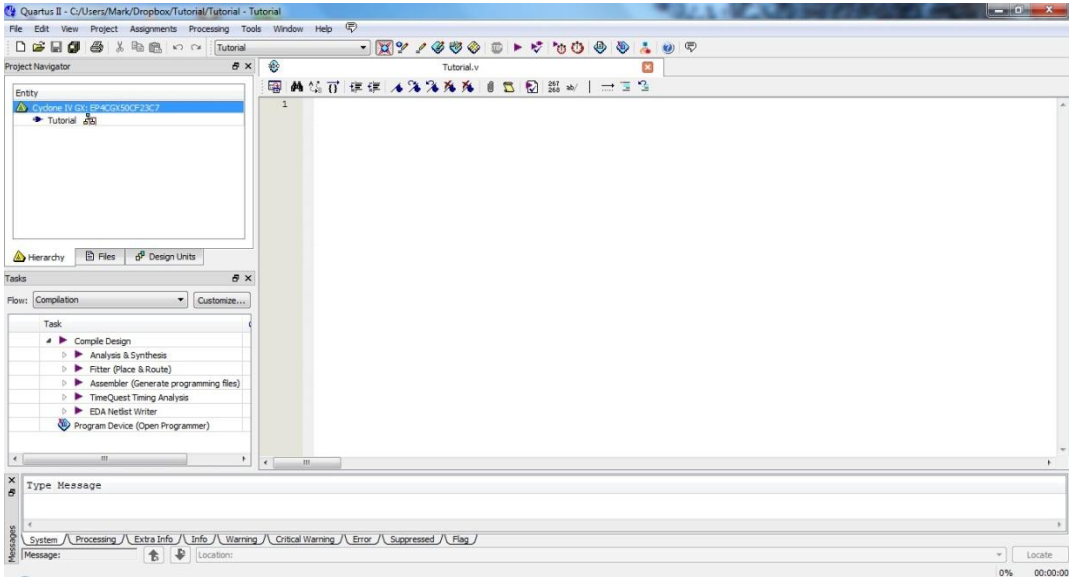
Device	EP4CGX15	EP4CGX22	EP4CGX30	EP4CGX50	EP4CGX75	EP4CGX110	EP4CGX150
Logic elements (LEs)	14,400	21,280	29,440	49,888	73,920	109,424	149,760
M9K memory blocks	60	84	120	278	462	610	720
Embedded memory (Kbits)	540	756	1,080	2,502	4,158	5,490	6,480
18-bit x 18-bit multipliers	0	40	80	140	198	280	360
PCIe® hard IP block	1	1	1	1	1	1	1
Phase-locked loops (PLLs) *(1)*	3	4	4	8	8	8	8
Transceiver I/Os	2	4	4	8	8	8	8
Maximum user I/Os	72	150	290	310	310	475	475
Maximum differential channels	25	64	109	140	140	216	216
Availability	Buy Now	Buy Now	Buy Now	Buy Now	Buy Now	Buy Now	Buy Now

Altera Tutorial

- Sidebar: What is a Cyclone II EP2C35F672C6?
  - We will not be actually doing putting your design on-chip, but other classes have attempted it, and some of you have experience in CSE 140L. We would like you to use the package denoted in Lab 2 for your testing.
  - The reason we want this is because, if you are arguing area or speed of the processor, the chip matters. If you are arguing something like cycle count, then this is less of a concern.

Altera Tutorial

- Step 6: Go to File-> New… and select Verilog HDL File to start coding up components. Be sure to save your files in the same directory as your project.



Altera Tutorial

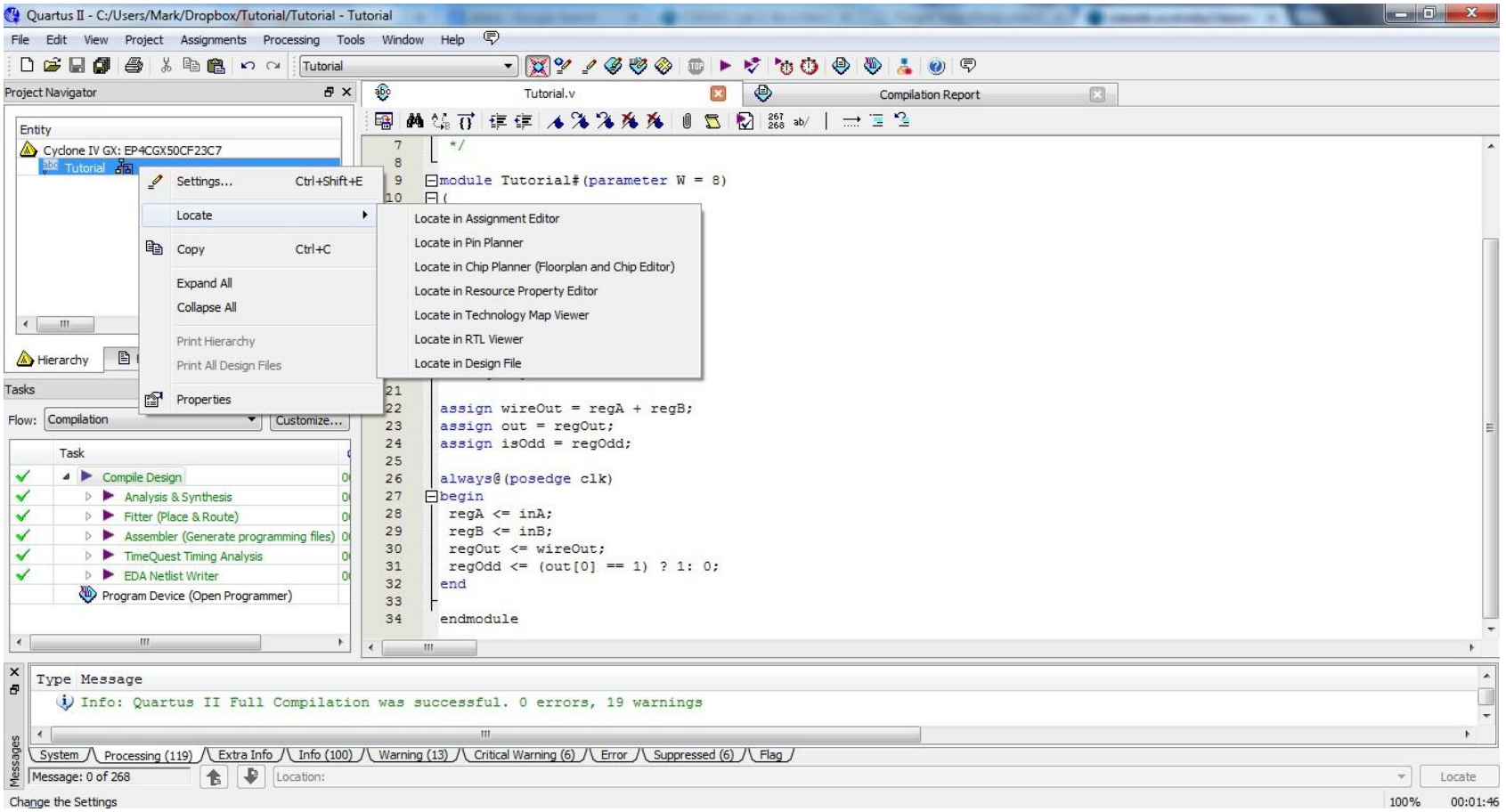
- Step 7: Insert the following code:

```
`timescale 1ns / 1ps

module Tutorial#(parameter W = 8)
(
 input clk,
 input [W-1:0] inA, inB,
 output [W:0] out,
 output isOdd
);

reg [W-1:0] regA, regB;
reg [W:0] regOut;
reg regOdd;
wire [W:0] wireOut;

assign wireOut = regA + regB;
assign out = regOut;
assign isOdd = regOdd;

always@(posedge clk)
begin
 regA <= inA;
 regB <= inB;
 regOut <= wireOut;
 regOdd <= (out[0] == 1) ? 1: 0;
end

endmodule
```

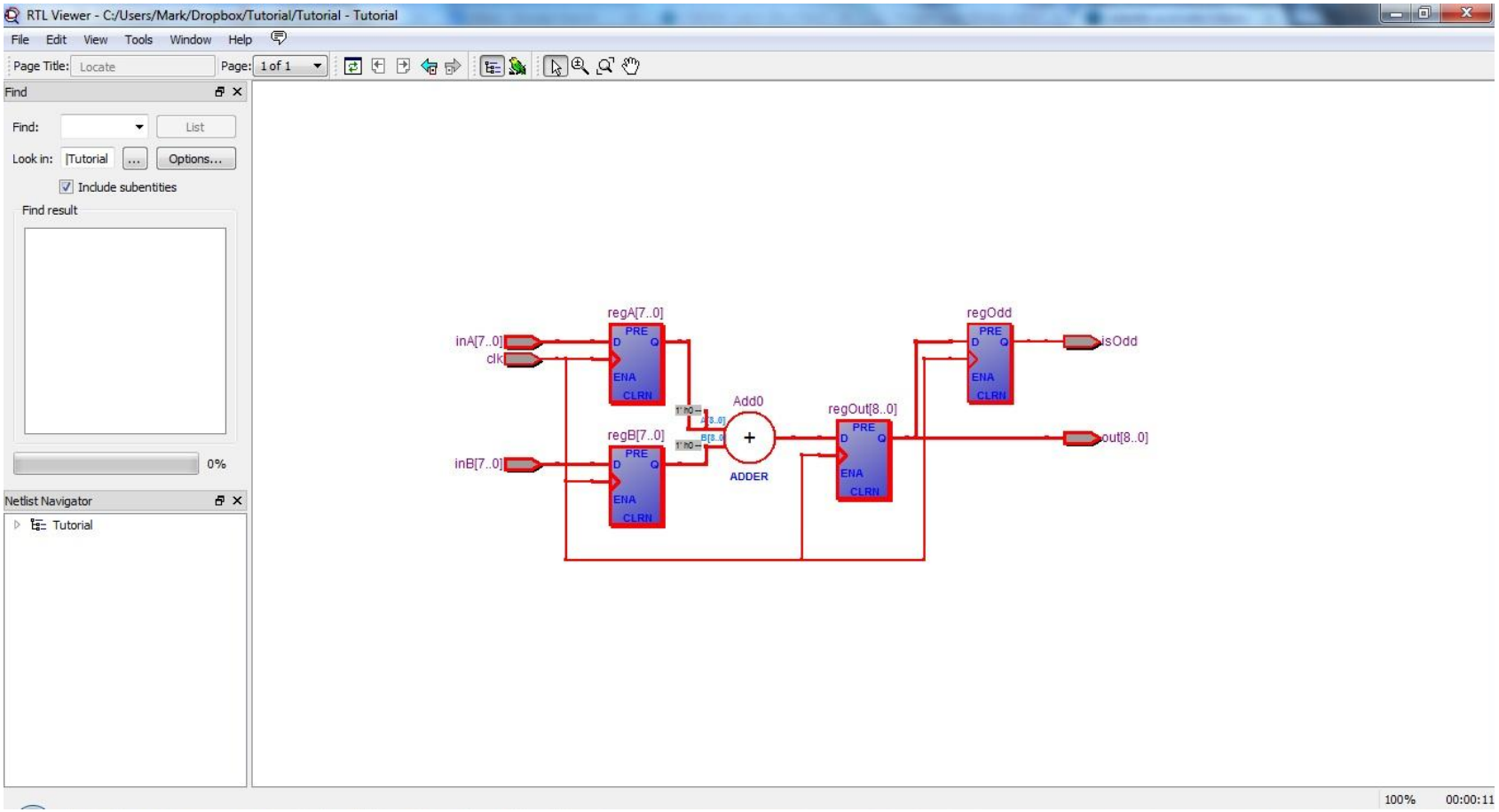
Altera Tutorial

- Step 8: Now its time to check to make sure everything is syntactically correct. Go to the tasks tab, and double click on “Analysis & Synthesis”. This will compile the code and show you any errors that may crop up in compilation.
- Step 9: If you want to see how things look at the RTL level, go to the hierarchy tab, right click your base module (in this case: Tutorial), go to Locate, and select “Locate in RTL Viewer”

Altera Tutorial



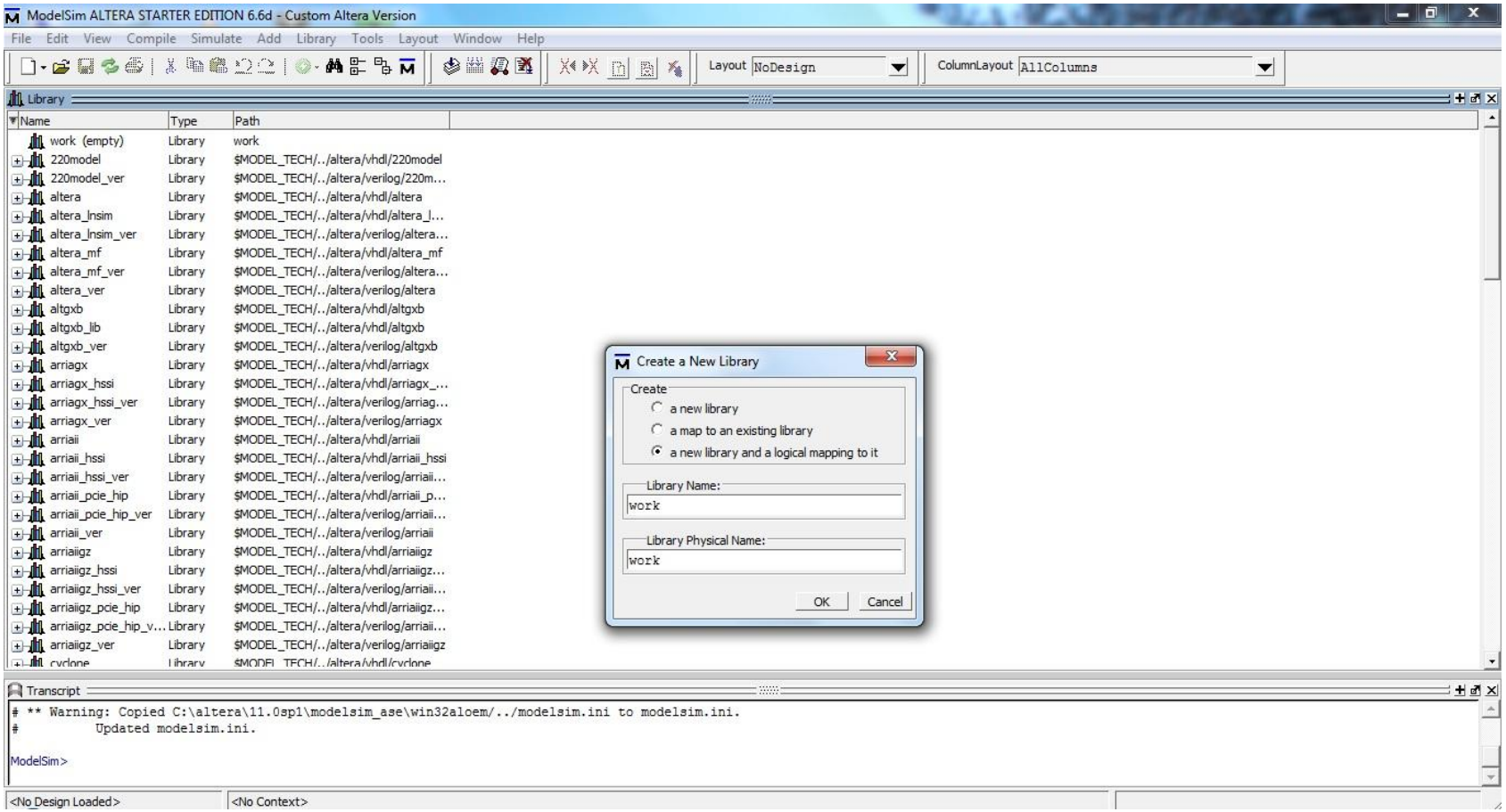
Altera Tutorial



Altera Tutorial

- Step 10: Repeat steps 6-8 for any other modules you wish to create. While we don't show it here, eventually you will have to wire up all your modules either using a schematic viewer or in verilog.
- Step 11: To see how it will work in simulation, we need to use modelsim. Open Modelsim up.
- Step 12: Go to File -> New… -> Library. Create a new library with the radio button "a new library and a logical mapping to it" selected.

Altera Tutorial



Altera Tutorial

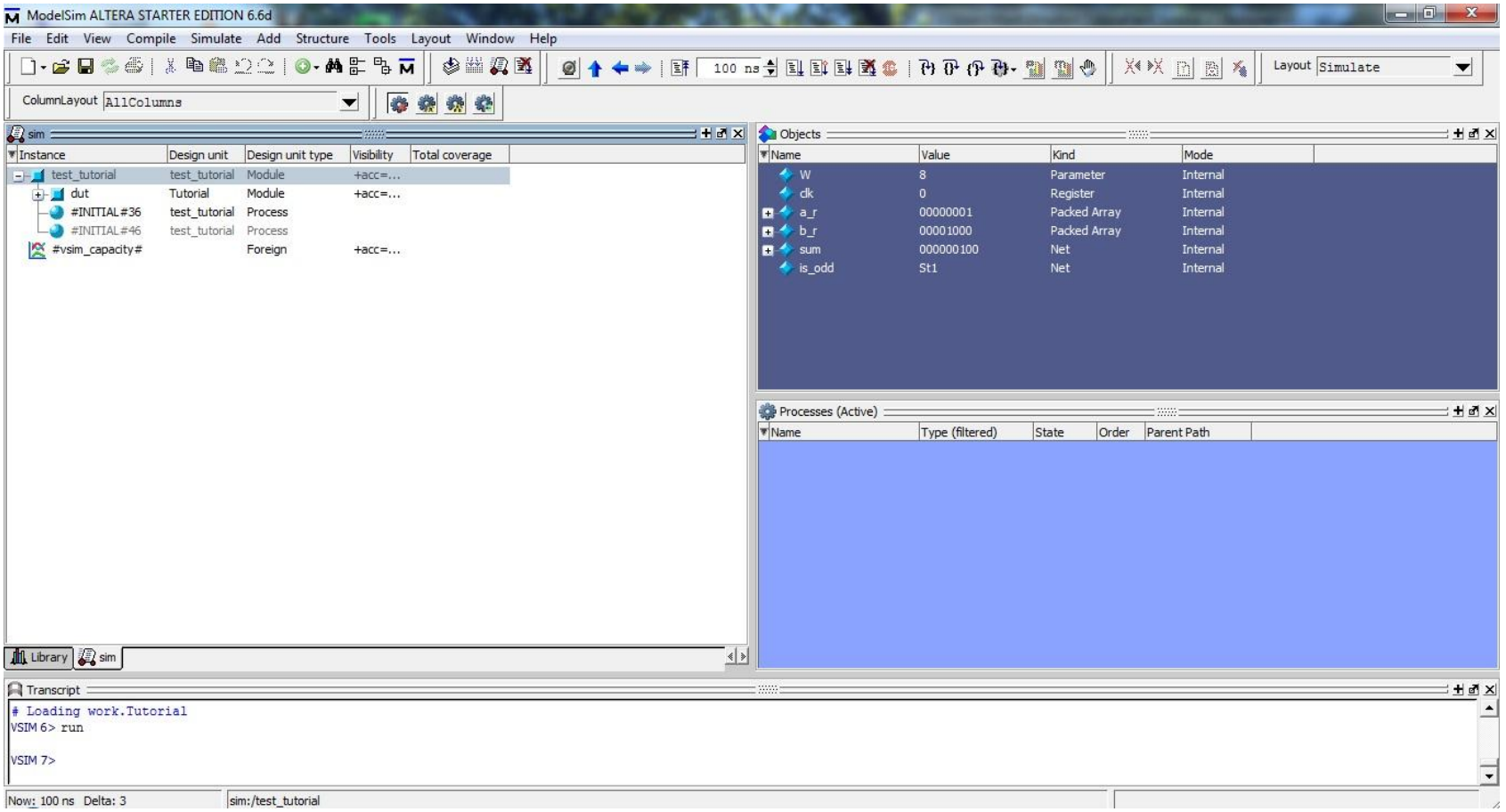
- Step 13: Create a file called “test_tutorial.v”, to set up the testbench (see next slide)
- Step 14: Go to compile->compile. Select BOTH the Tutorial.v and test_tutorial.v files to compute.
- Step 15: Go to your work library and double click “test_tutorial.v”
- Step 16: Go to the top bar, change 100ps to 100ns, DO NOT hit run yet.

Altera Tutorial

```
`timescale 1ns / 1ps

module test_tutorial#(parameter W = 8);

   reg          clk;
   reg  [W-1:0] a_r;
   reg  [W-1:0] b_r;
   wire [W:0] sum;
   wire         is_odd;

   // The design under test is our adder
   Tutorial dut (   .clk(clk)
                        ,.inA(a_r)
                        ,.inB(b_r)
                ,.out(sum)
                ,.isOdd(is_odd)
              );

   // Toggle the clock every 10 ns

   initial
     begin
        clk = 0;
        forever #10 clk = !clk;
     end
```

```
// Test with a variety of inputs.
   // Introduce new stimulus on the falling clock edge so that values
   // will be on the input wires in plenty of time to be read by
   // registers on the subsequent rising clock edge.
   initial
     begin
        a_r = 0;
        b_r = 0;
        @(negedge clk);
        a_r = 1;
        b_r = 1;
        @(negedge clk);
        a_r = 5;
        b_r = 6;
        @(negedge clk);
        a_r = 2;
        b_r = 2;
        @(negedge clk);
        a_r = 3;
        b_r = 3;
        @(negedge clk);
        a_r = 1;
        b_r = 8;
     end // initial begin

endmodule // test_adder
```

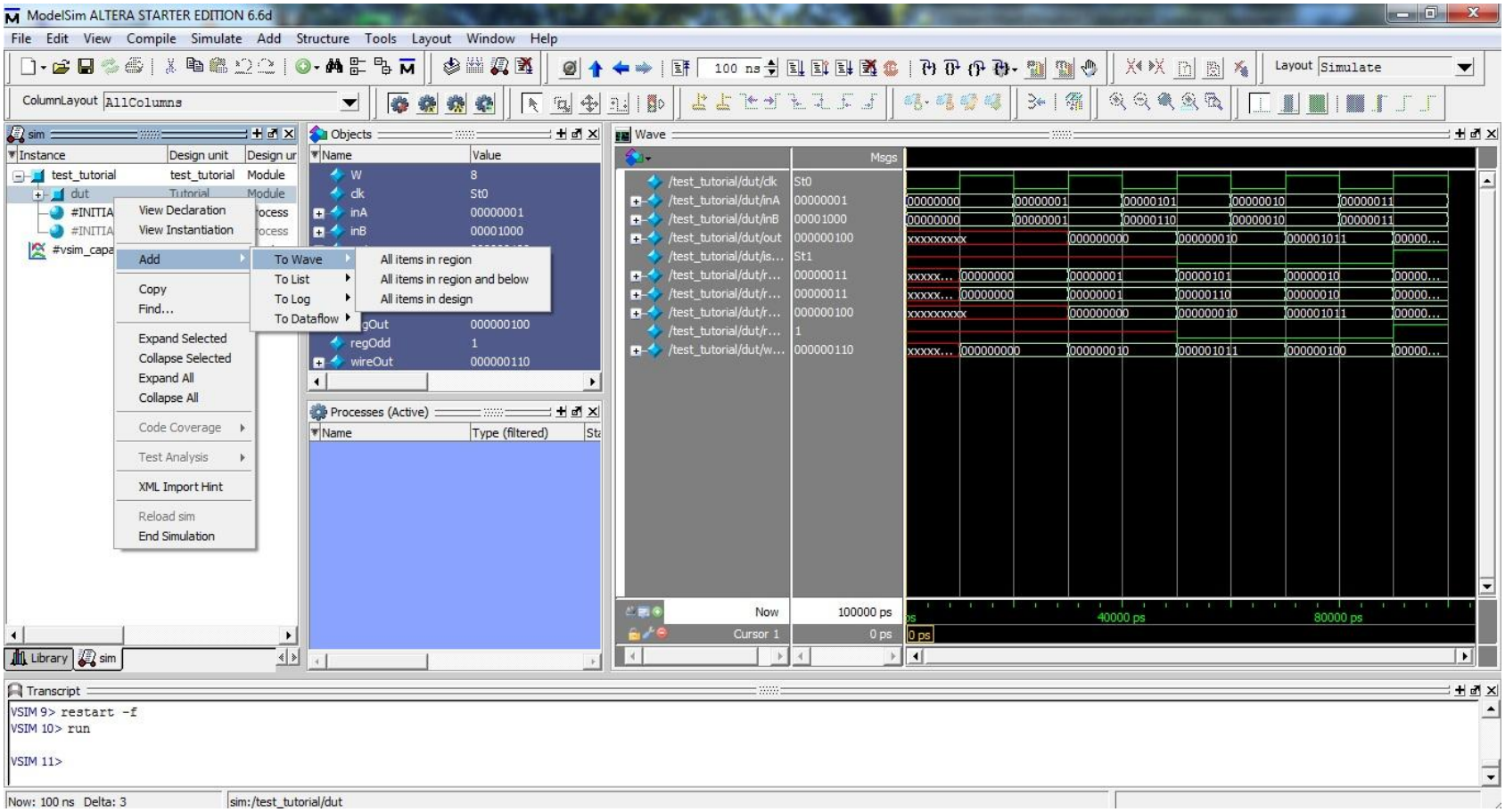
Altera Tutorial



Altera Tutorial

- Step 17: To see the waveform, right click “dut”, go to Add -> To Wave -> All items in region
- Step 18: Now hit run. Right click the waveform and select “Zoom Full” (or press F).
- Step 19: If you did everything properly, the waveform should be displayed as shown here.
- Step 20: Note that the adder has a delay between the input and the correct output. This was by design. Be careful when coding in verilog not to have too many registers in your modules as this can slow down your processor.

Altera Tutorial



QUESTIONS?