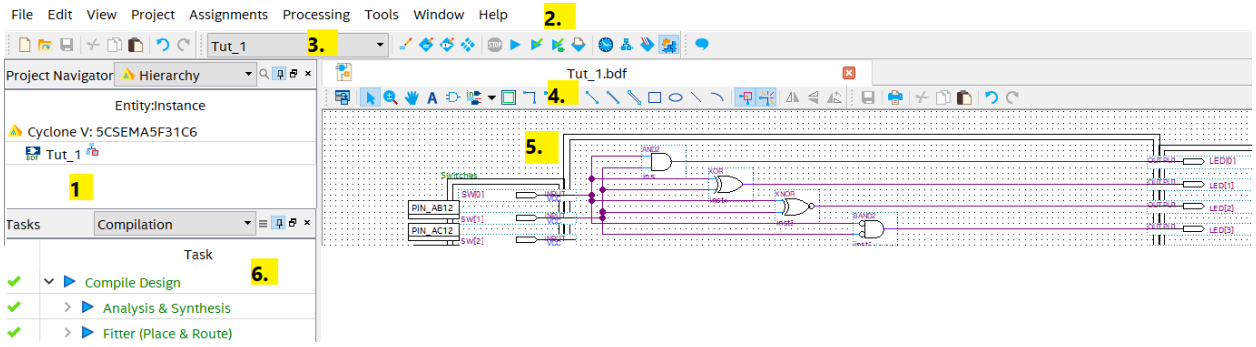
1) Schematic Design

- Download Quartus project and setting files: **Lab_Template.rar** or **Lab_Template.zip**
  Extract ALL files in the folder to your working directory.
  - Save a copy of the project templates somewhere for your future labs.
  - Start **Quartus,** then **open project ->** select the **Tut_1.qpf** file in your working directory. Double click on the Tut_1 file to open the schematic template for your design entry.



1. Project Navigator
2. Command Bar
3. Icon Bar
4. Workspace Toolbar

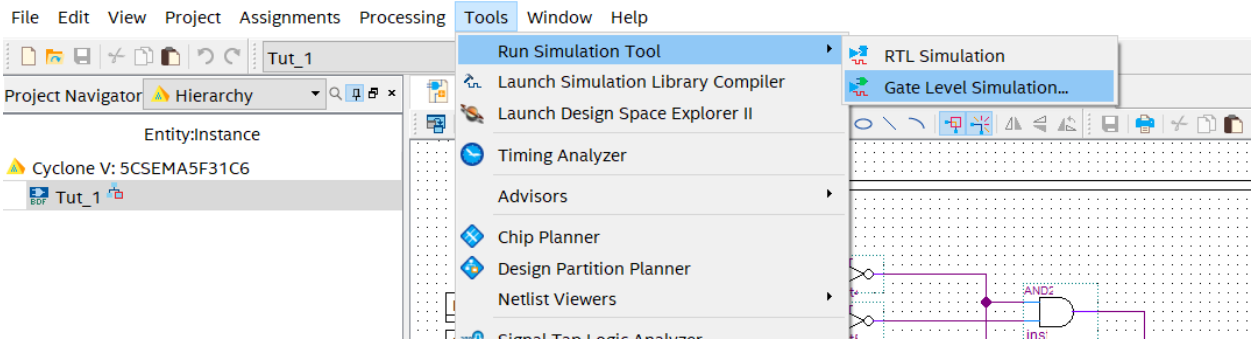
For most of the laboratory experiment the "primitive" symbols will be used. The orthogonal node tool allows the user to place a virtual wire between different inputs and outputs.

5. Workspace
6. Taskbar

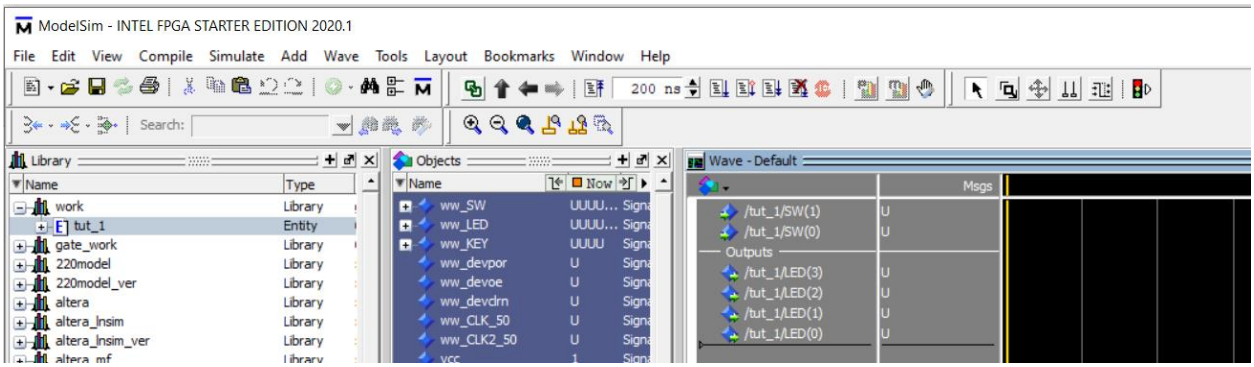
By clicking on the "play" button the design will go though the stages to create the programming document.

2) Simulating a Design

1. Launch ModelSim



2. ModelSim should now be launched. Locate the "**Library**" window. Locate the "**work**" file (it should be the top file). Double click on it to see its contents and double click "tut_1".



3. In the center column should be an "**Objects**" window. Select the ones you used for your schematic (e.g. SW0, SW1, LED1…) and click "**Add Wav**e".

4. The set of waves should now be in the simulation window. Now values must be assigned to them so that a simulation takes places. To do this we make each input a clock cycle with different periods.

5. Before we simulate the design, we must ensure that the simulation length is adequate.

6. By clicking on different time points in the simulation, the values of the inputs and outputs can be read off

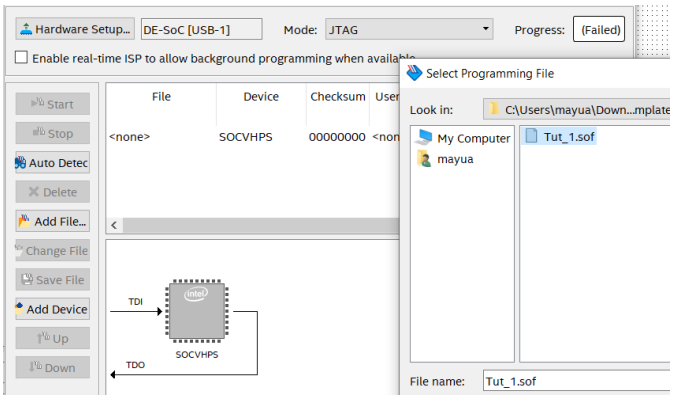
3) Implementation

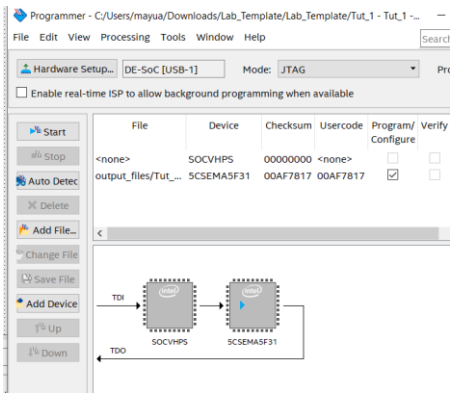
1. Connect the power of the De1-SoC board; connect the USB from the DE1-SoC board to the computer. Turn on the board.

2. In the "**Tasks**" bar select "**Program Device (Open Programmer)**"

3. Make sure "DE-SoC[USB-1] is correctly recognized, and Mode is JTAG. Use "Auto Detect" to detect the board, and then add the Tut_1.sof program file from the "output" folder of your project.

4. Click "Start" to program the board.





4) Laboratory Exercises

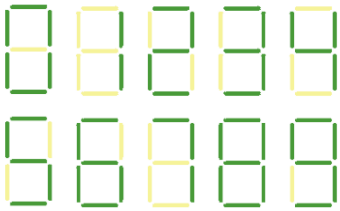
With an understanding of the design, simulation, and implementation of the binary several exercises can be done. There are two exercises, one requires repeating the AND2 design with other primitive gates and the other is a binary to seven segment display.

a) Logic Gates

Repeat the steps taken with the AND2 gate with the XNOR, XOR, BAND2 gates and record the results.

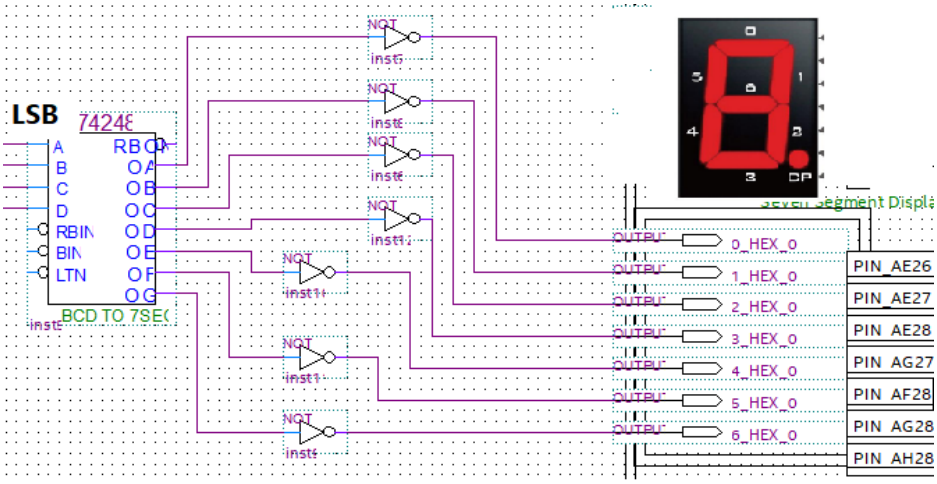
b) Seven Segment Display

The two white square boxes on the top left of the Binary Explorer is a seven-segment display. Each segment of a seven-segment display is a small light-emitting diode (LED), and - as is shown below - a decimal number is indicated by lighting a particular combination of the LED's elements. Binary-coded-decimal (BCD) is a common way of encoding decimal numbers with 4 binary bits:



Decimal digit	0	1	2	3	4	5	6	7	8	9
BCD code - $D_3\ D_2\ D_1\ D_0$	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

In the second part of this lab, a test circuit will be constructed to drive the 7-sgment display using a 4-bit BCD signal. Press "**Add Symbol**" and chose the "**others**" section, then press "**maxplus2**" and pick the **74248** chip. Connect the four switches and the outputs to their respective segments as shown. Please note the segments are active-low (have common anode, a '0' turns the LEDs on), so seven inverters need to be added for the 7-segment display. Then compile and implement (no need to simulate) the design and record the results.



Digital Circuits - ECED 2200 Tutorial 1 Observations

Student Names: _____ B00_____

AND Gate Truth Table:

A	B	Y
0	0	
0	1	
1	0	
1	1	

XNOR Gate Truth Table:

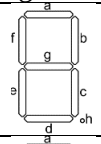
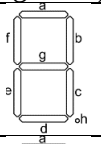
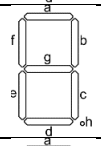
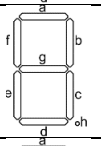
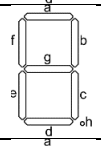
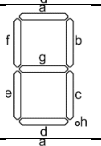
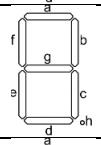
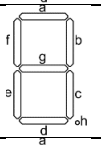
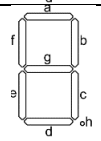
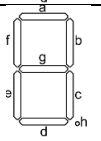
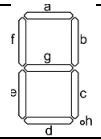
A	B	Y
0	0	
0	1	
1	0	
1	1	

XOR Gate Truth Table:

A	B	Y
0	0	
0	1	
1	0	
1	1	

BAND2 Gate Truth Table:

A	B	Y
0	0	
0	1	
1	0	
1	1	

Switches $SW_4\ SW_3\ SW_2\ SW_1$	Display (colour the light-up segments)	Switches $SW_4\ SW_3\ SW_2\ SW_1$	Display (colour the light-up segments)
0000		0110	
0001		0111	
0010		1000	
0011		1001	
0100		1010 (what happens @ 10?)	
0101		1011