EEE 3530
Spring 2020

Project #1-1: Combination Circuit Review on Quartus II

Objectives:

To learn how to program and build your own simulation using Quartus

Due Dates:

Softcopy: April 14th, 6PM (To YSCEC)

** Late submission within 24 hours automatically deducts 10 percents of the point.*

Things to Submit:

1. *Report*
   Cover page with your name and Student ID
   Schematic diagram from Quartus II (your design)
   Simulation results (the wave form)
   Discussion

2. *Your design files*

* *Working out must be shown*
- Please match the format *Name(Korean)_StudentID.zip*

Professor: W. W. Ro

TA: Jonghyun Lee, Jeonghoon Choi

Feel free to contact us whenever you meet some difficulties or errors in the manual (but, spend some time and efforts to figure them out for yourself first)

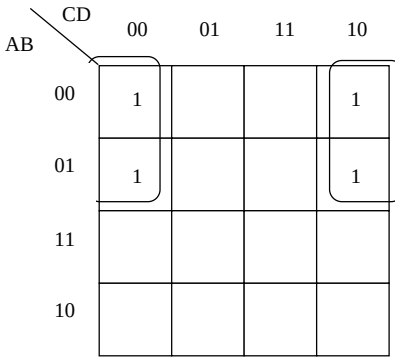
Combinational Circuit to Detect Any Single Digit in Your Student ID

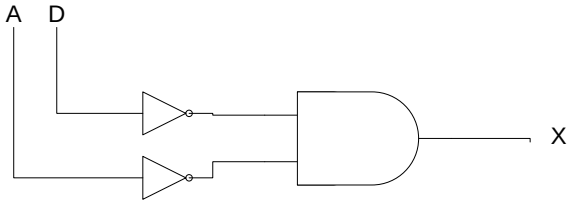
The first design project is to implement a simple combinational circuit using traditional gates you learned from "Digital Logic Circuits" First, fill out a *truth table* and the *Karnaugh map*, and implement the circuit (everything on paper, first). Second, implement the same circuit using the *schematic capture* on Quartus II software. **This tutorial is based on version 9.0 Web Edition**. (*visit http://www.altera.com to find the software*). **You should choose version 9.0** since 10.0 or later would not have vector waveform editor.

Design requirements: The input is four bit signal (four inputs as a single bus) which represents an unsigned binary number (Ex: 0001 → 1, 1001 → 9, 1111 → 15). If the input signal represents any single digit in your Student ID, the output signal produces 1. It is a simple combinational circuit that you can implement easily based on the "Digital Logic Circuit" material (Yes, it is an easy circuit; the purpose of this project is mainly to be familiar with the tool). You can first follow the procedure described in this manual. It implements the circuit for Chulsu's ID (example ID).

EX) Assuming that Chulsu's ID is 0240206 → the output becomes 1 when the input is 0, 2, 4, or 6. Below are the truth table, K-map, and implementation; those are Chulsu's homework. You need to do with your Student ID.

Input				*Output*
A	***B***	***C***	***D***	***X***
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0



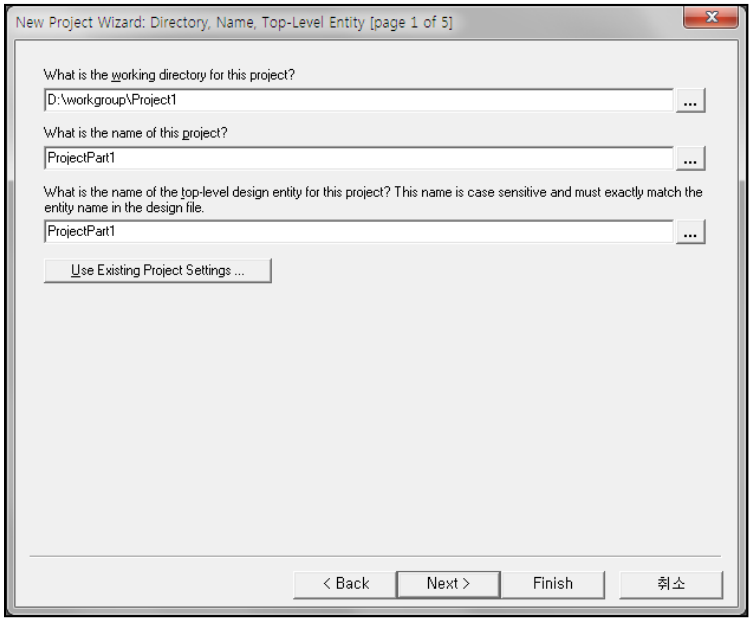


F (A, B, C, D) = A′D′

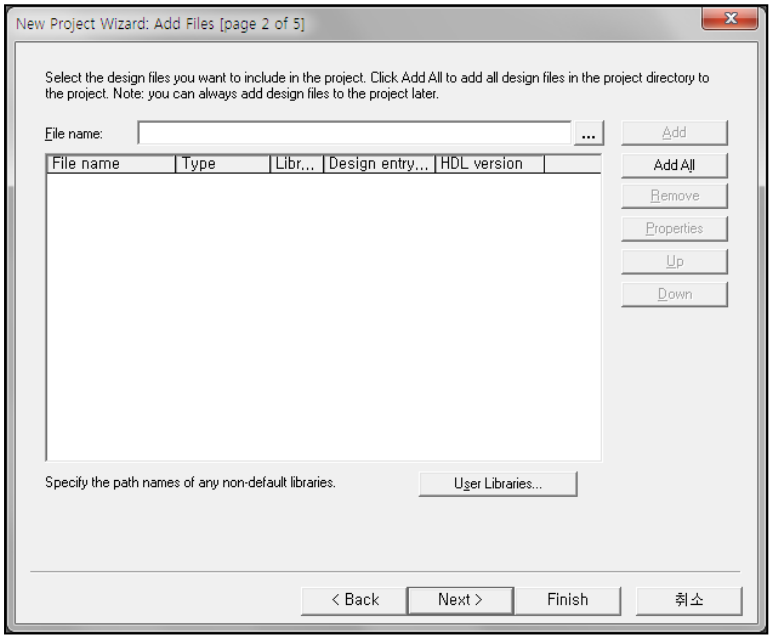
Now, we can draw the above circuit and simulate the circuit using the Quartus II software. Start the software first. (First, click Quartus II icon on Desktop to start the program).

A. Set Up the Configurations

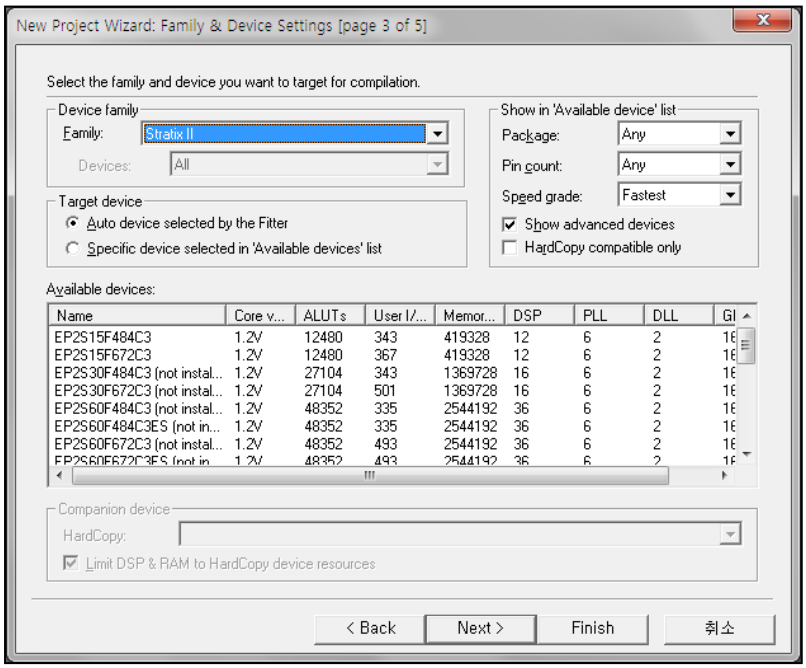
1. Go to "File" → "New Project Wizard"
2. Enter your project directory (whatever directory you want to save the project in) and project name (Important! Your project name should include your initial). Also, make the name of the top-level entity same as the project name. Click "Next"



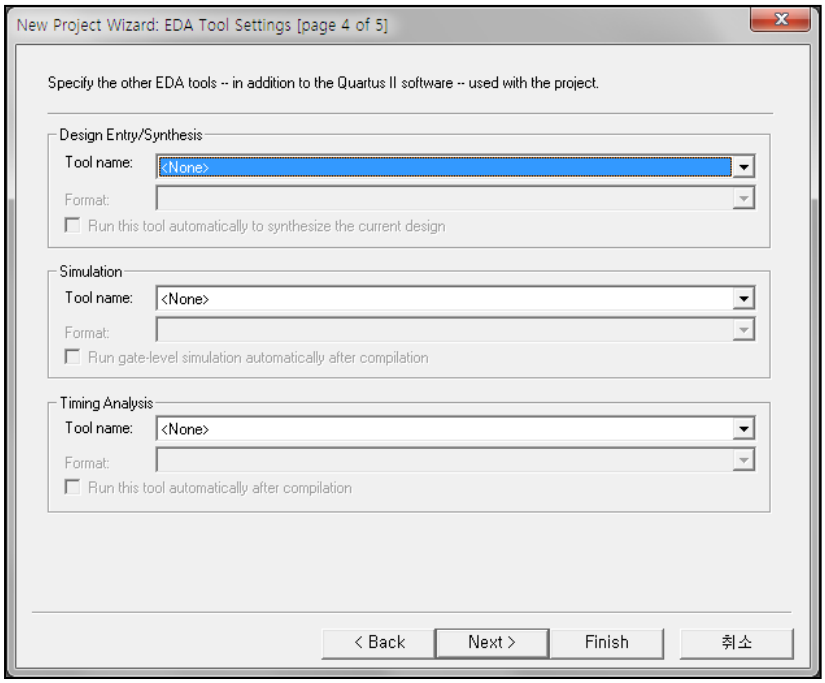
3. In the following window, just click "Next"



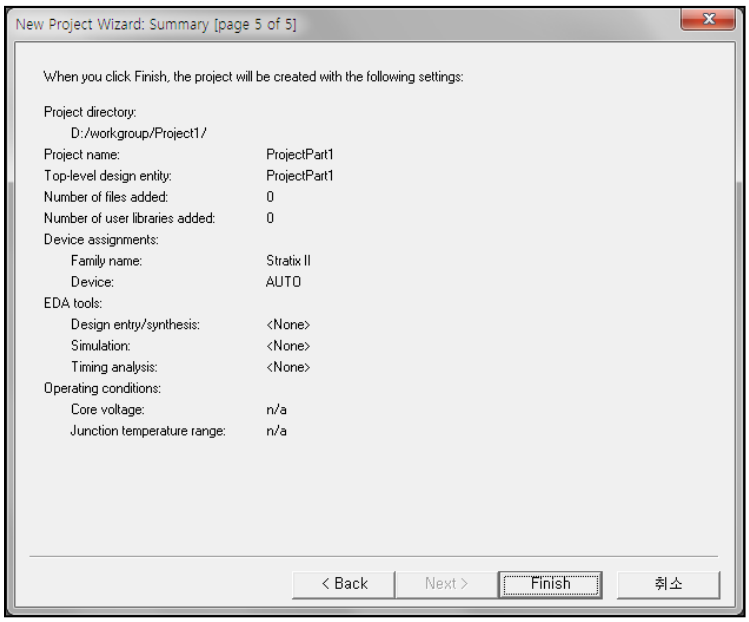
4. You will see the following diagram; just select the same options shown in the following diagram. It is not so important since we don't use any device in this course. Click "Next".



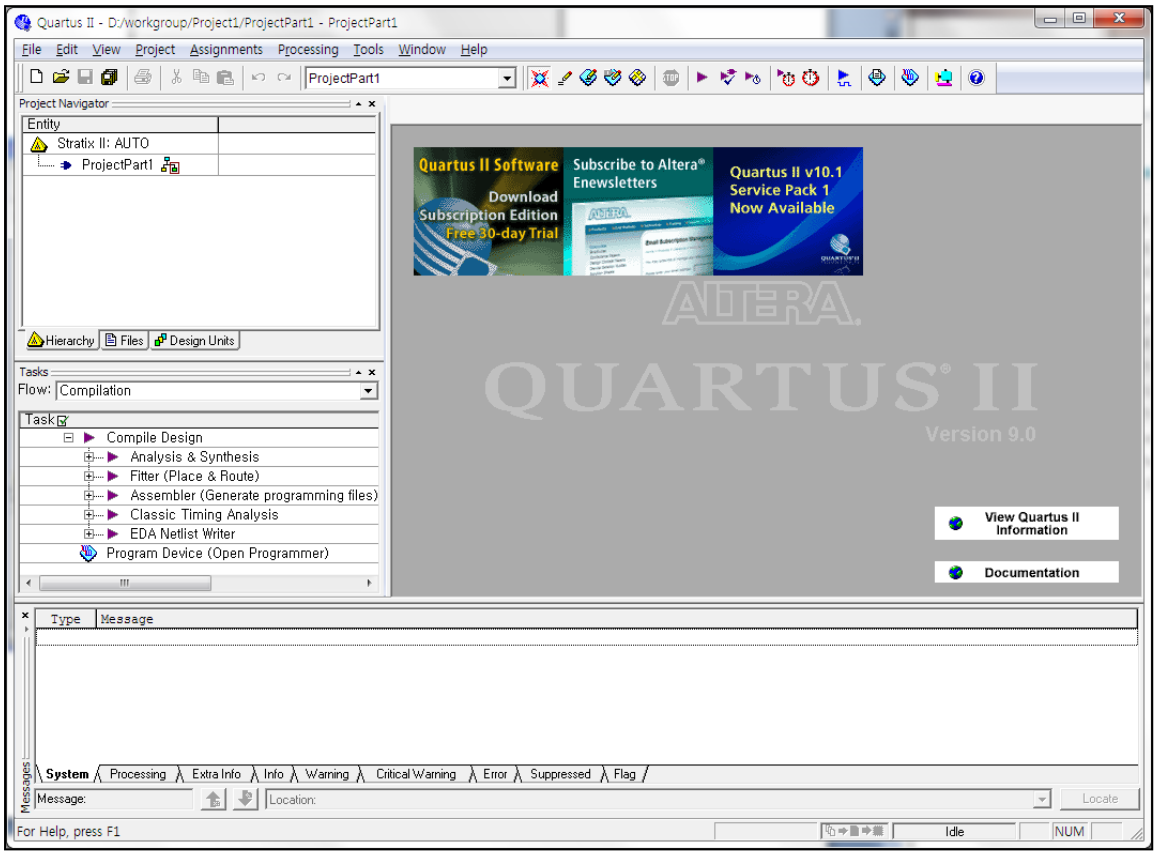
5. Then, you will see the following box pop up. Just click "Next".



6. Select “Finish”.

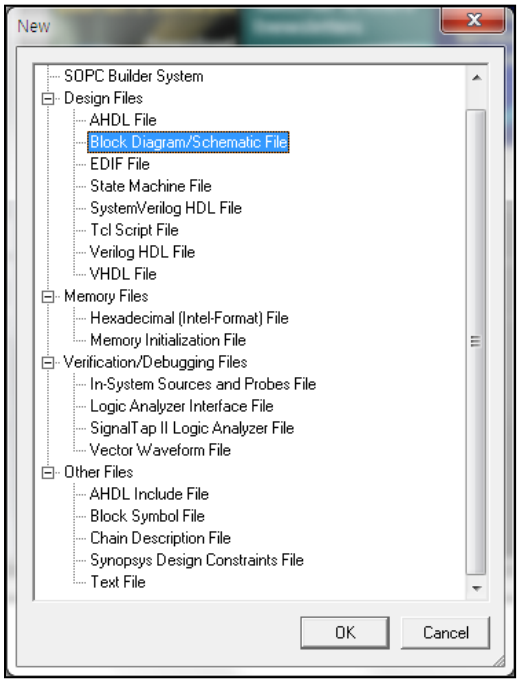


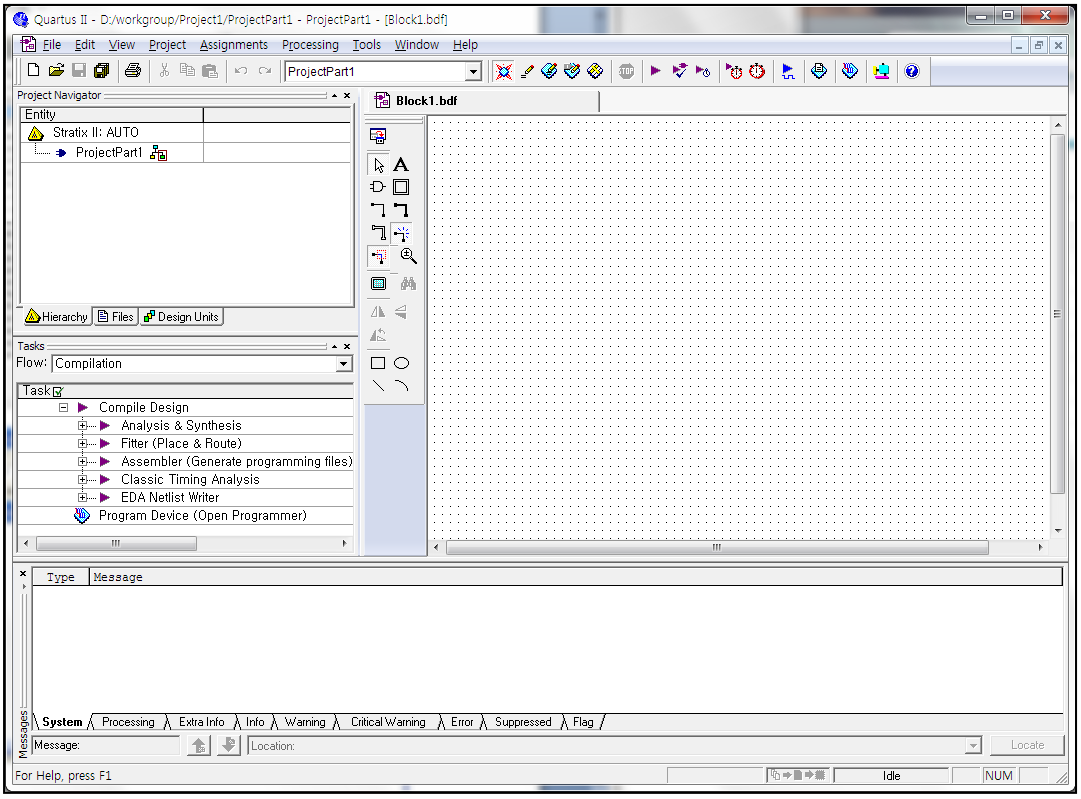
7. Finally, we will get to the following window. Now, let’s start drawing the gates and wires.



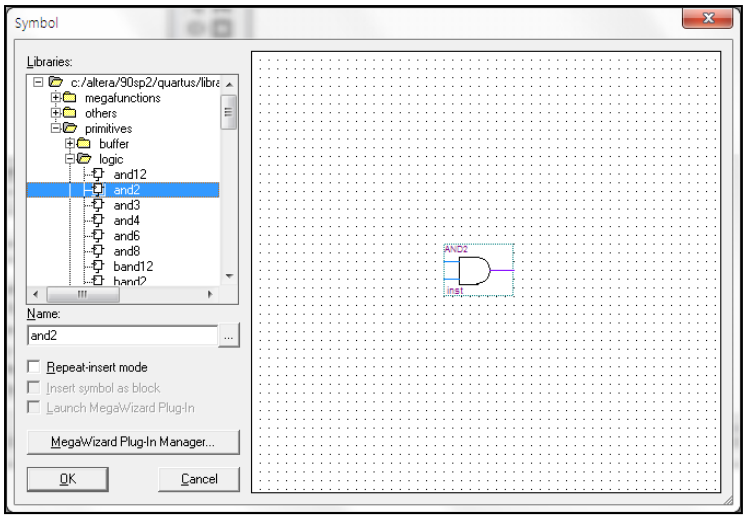
B. Schematic Drawing

1. First, go to "FILE → NEW", and the following pop-up menu will appear. Select Block Diagram/Schematic File", then click "OK"

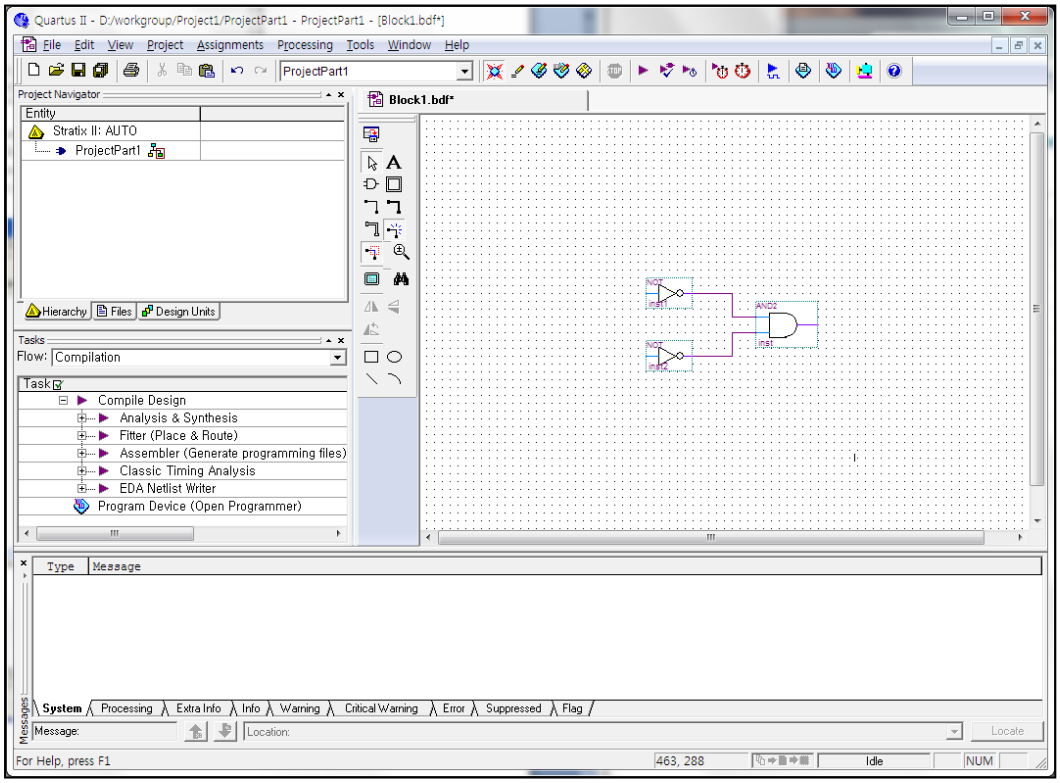




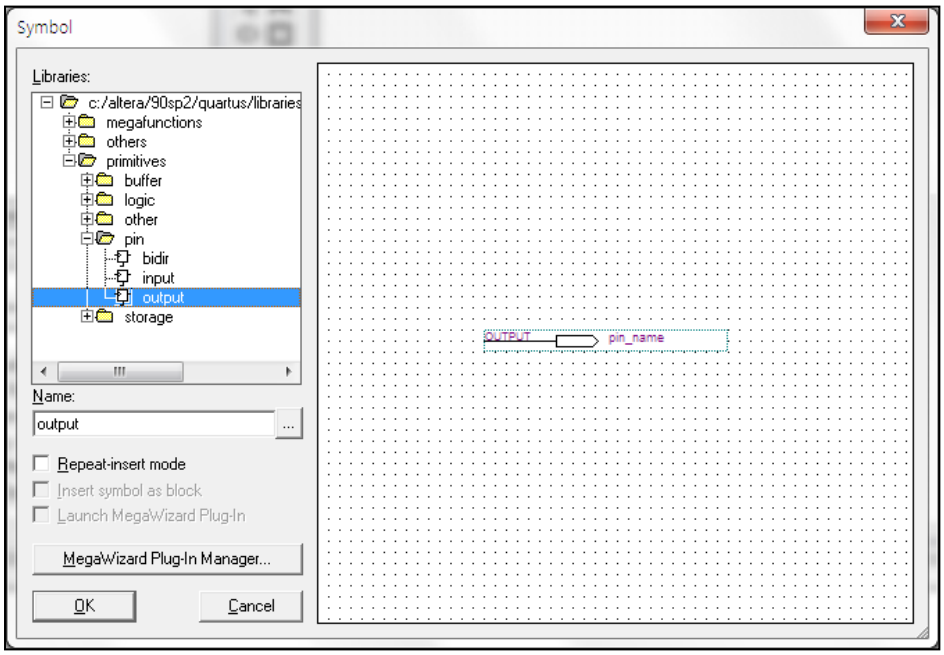
2. Double click on any point in the dotted area (or you can click the AND gate symbol on the tool bar). You will see the following window pop up. Expand by clicking the folder "C:/altera/90sep2/quartus/libraries". It may show different path depending on your configuration. Then, in the folder tree, expand "primitives". Finally, you can expand "logic" and find "and2" for 2-input AND gate.



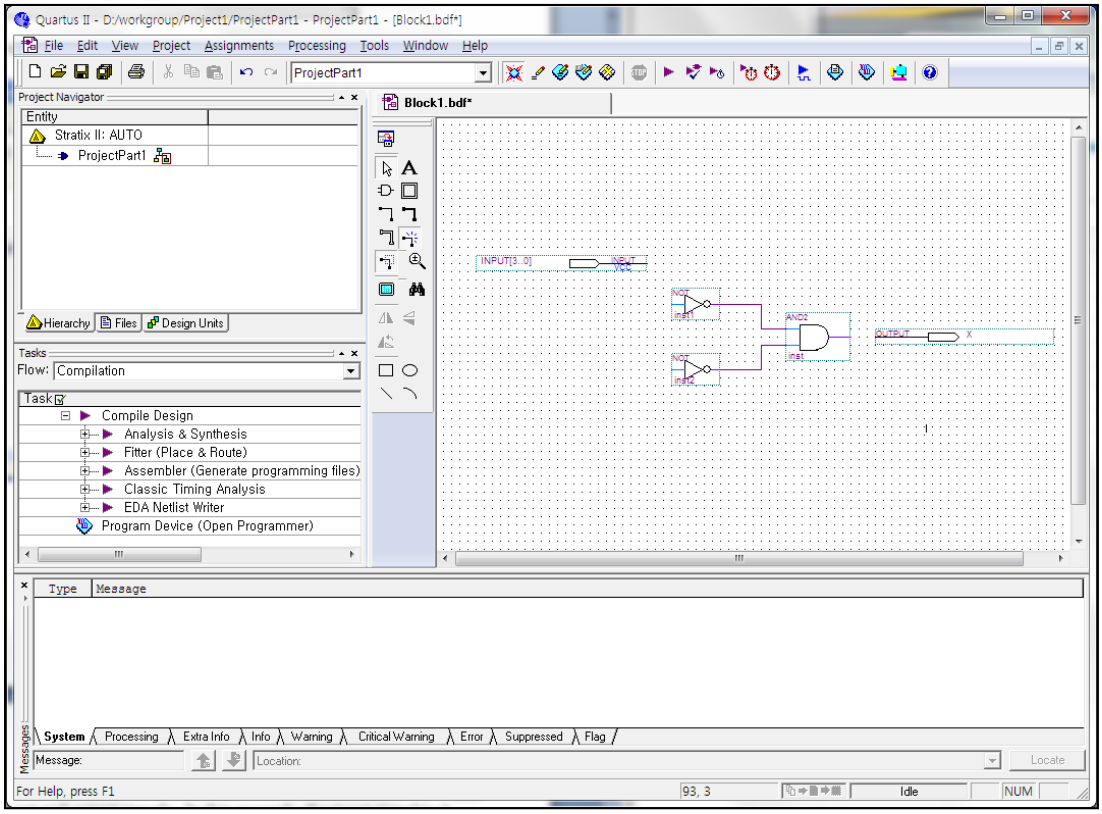
3. You can find any element (Symbol) you need to draw for the schematic. In our example, we need a two-input AND gate and two inverters. Click each element at one time and draw the circuit. To connect each element, you can just click one element and drag it to others (or you can use the Orthogonal Node Tool; below the AND symbol on the tool bar)



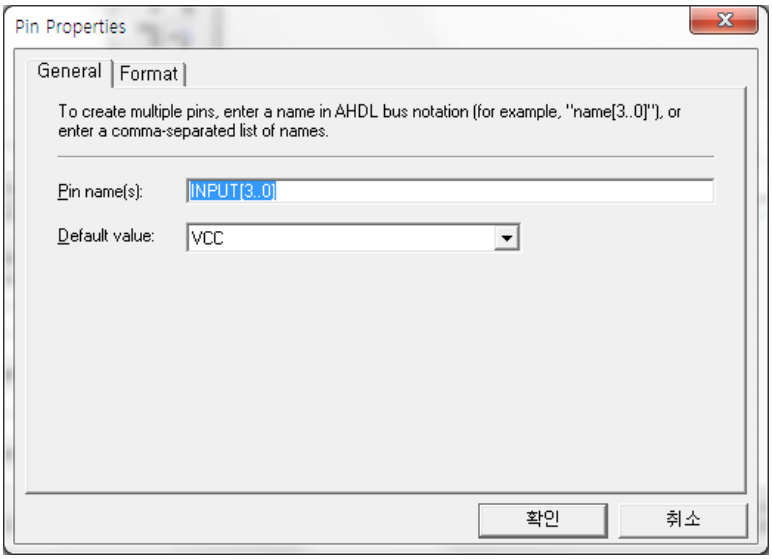
4. Now, we need to insert input and output. This time, expand the pin library, select input, and click OK. Do the same procedure again, to insert output pin also.



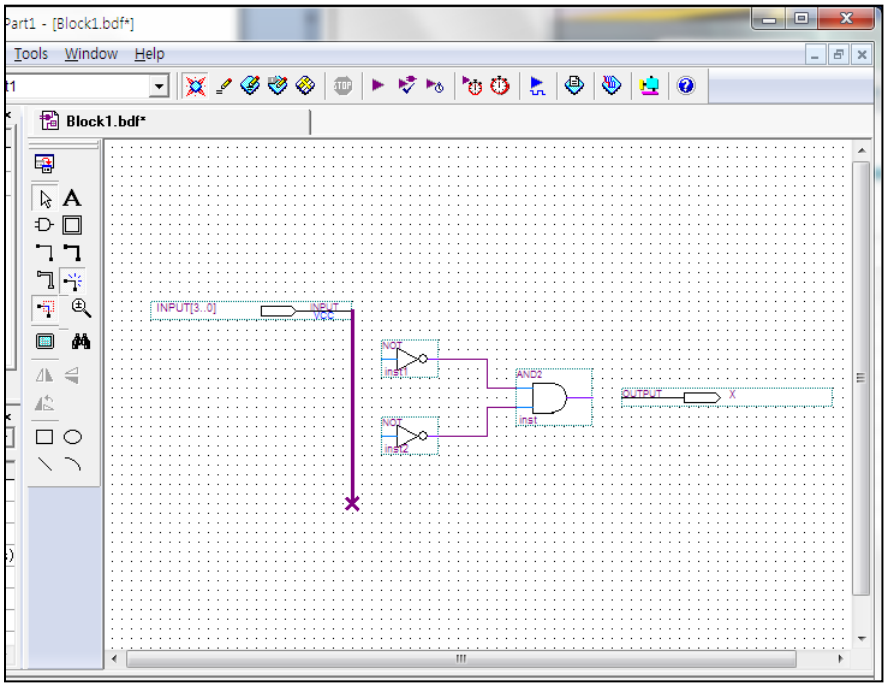
5. Now, place those elements such as in the following schematic.



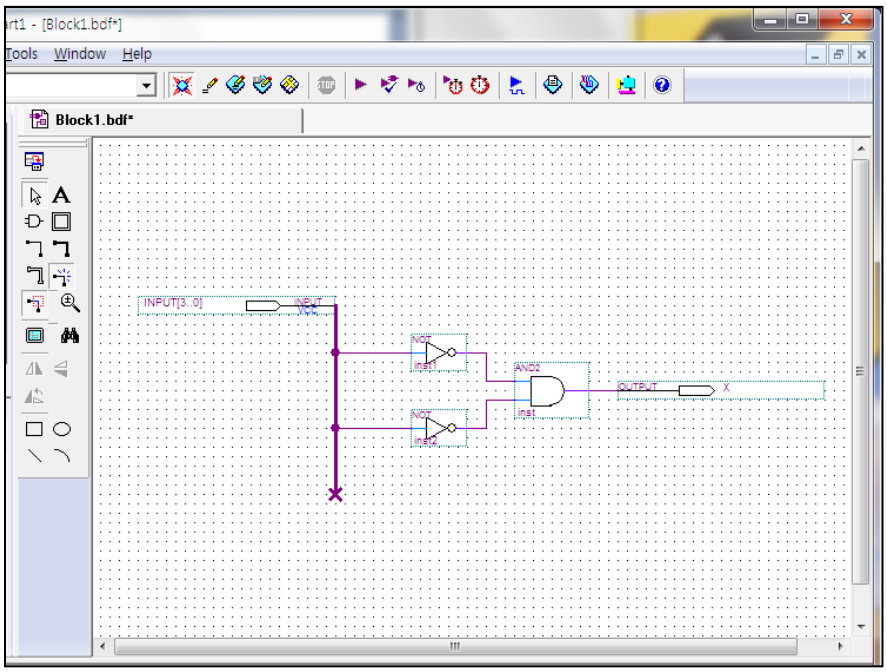
NOTE: We also need to name each input and output signals. In this example, the input signal is 4-bit binary number. First, click right-button on the input symbol and select "Properties". In the following box, type INPUT[3..0]. Follow the same procedure for the output and name the output as X. The circuit has only one output.



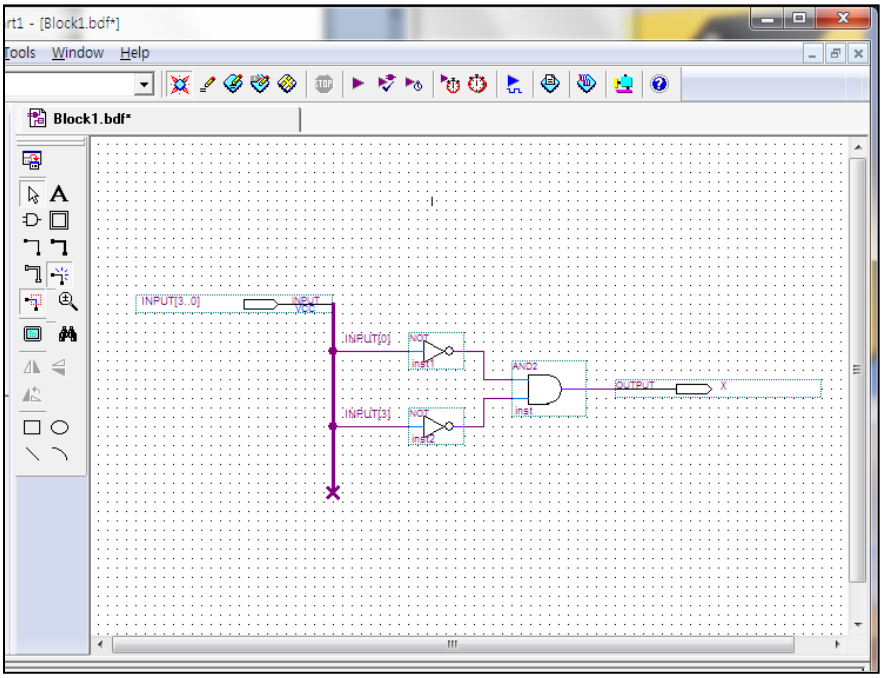
6. The last step is connecting each element. First extend the input signal by using the Orthogonal Bus Tool (or you can just place mouse point at the right side of the input wire and drag).



7. Now, connect each inverter to the bus. Also, connect the output of the AND gate to the output.



8. Now, inputs for two inverters are supposed to be connected to the bus; however, it does not recognize which one signal of the four inputs comes to. Therefore, you need to name it explicitly. Right button click on each input of the inverter and go to "Properties". Name each input as "INPUT[0]" and "INPUT[3]" as seen in the following diagram.

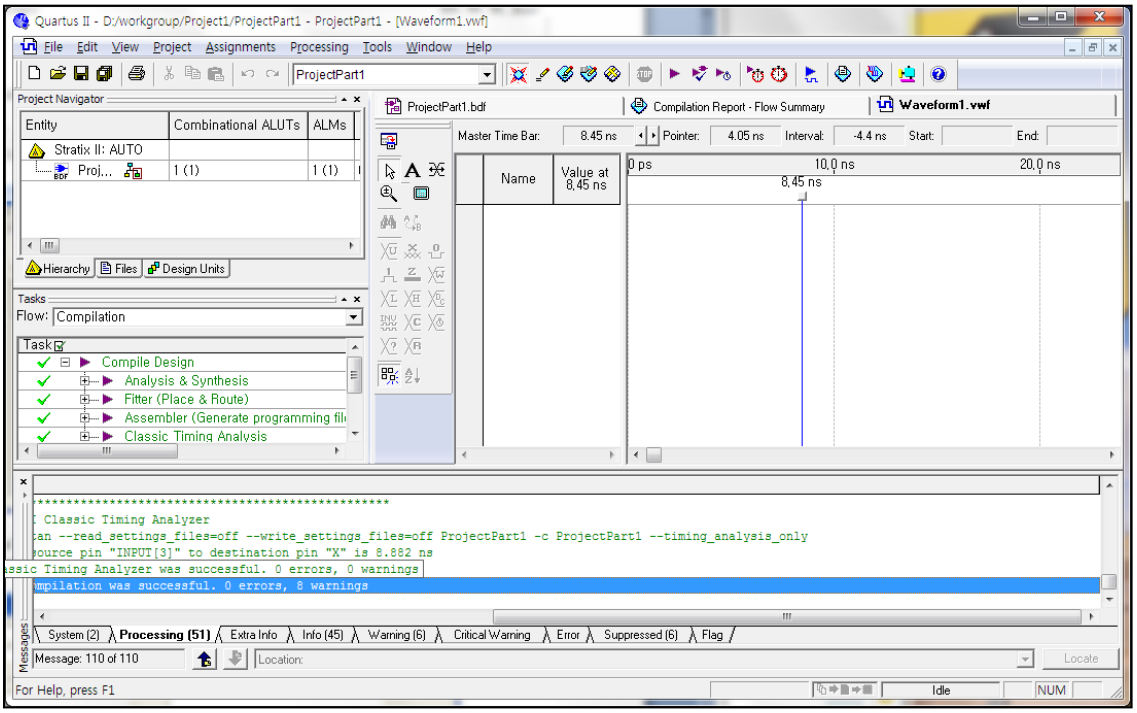


9. All right! Now, go to "Processing" → "Start Compilation" to check whether your design has any errors. If it shows "Successful", your design is correctly implemented. If you encounter any error message, debug your design.

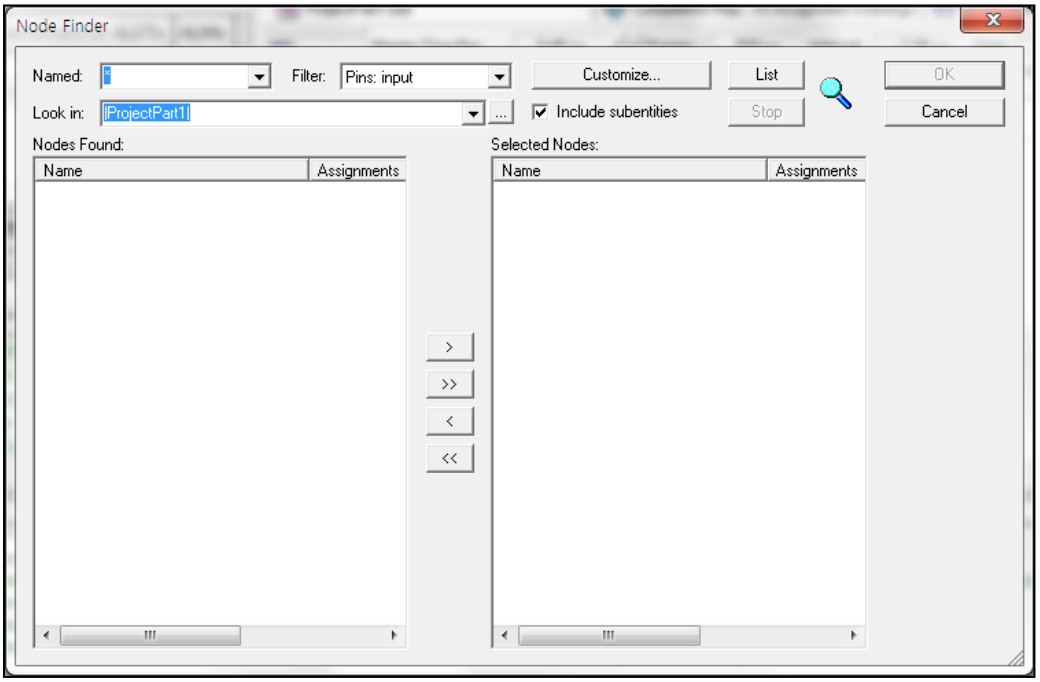
C. Logic Simulation

Now, your circuit is ready and you can simulate it with the software.

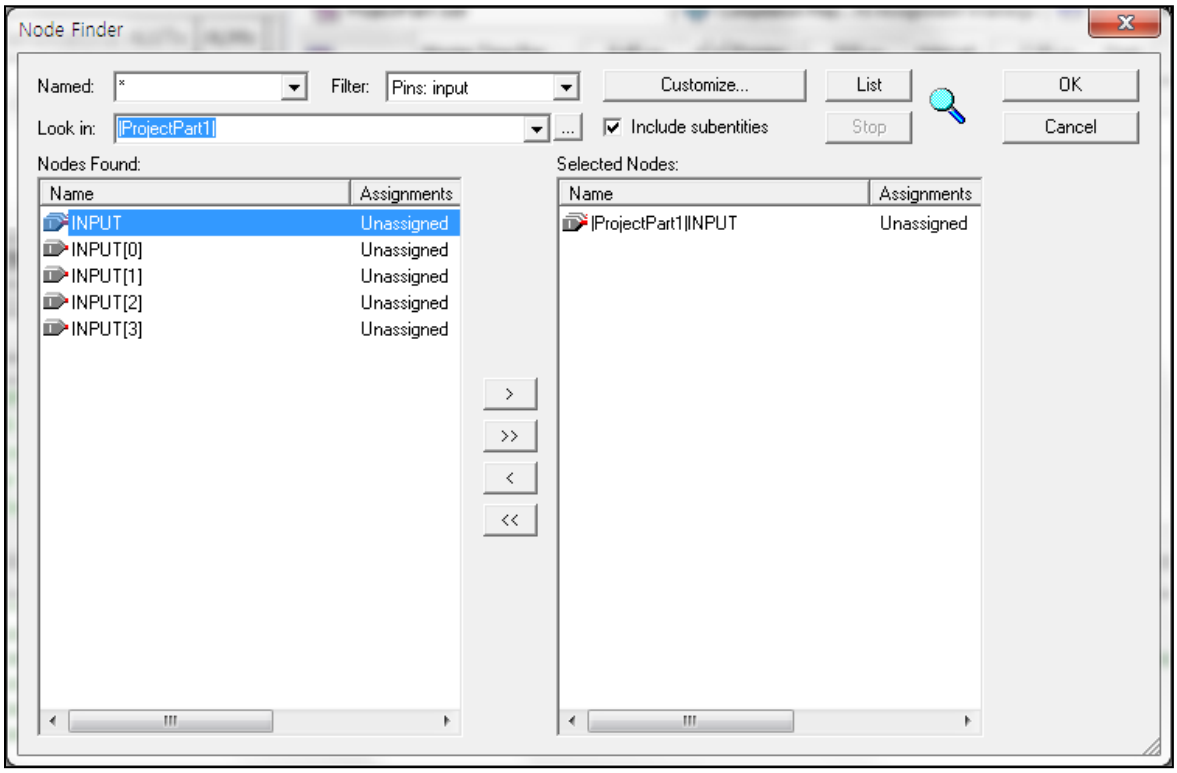
1. Choose "File" → "New", and then from the popup window, select "Vector Waveform File" and click "OK". You will see the following blank waveform window.



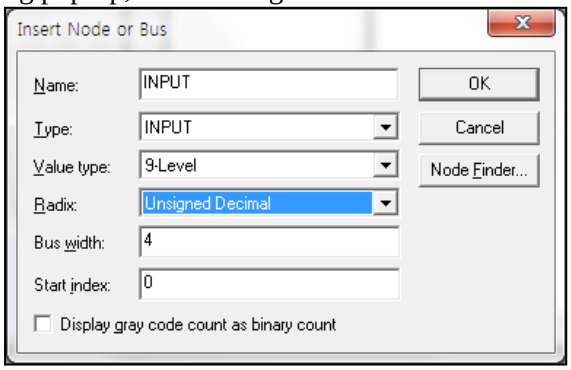
2. Double click on the "Name" column on the left side. Select the "Node Finder". Also, click "List".



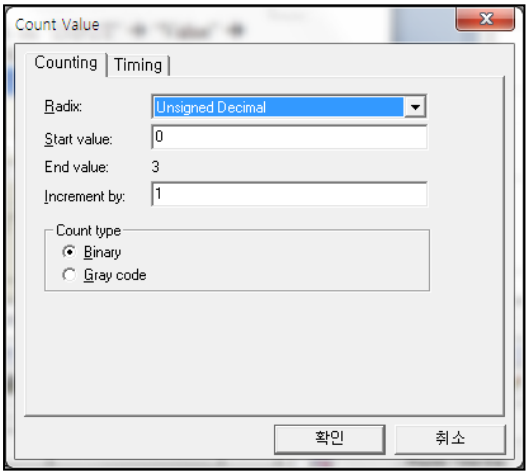
3. Right click on the "Name" column on the left side. Select "Insert Nodes or Bus". Click on the "Node Finder". Also, select "Pins : input" and click "List". Then Select "INPUT".

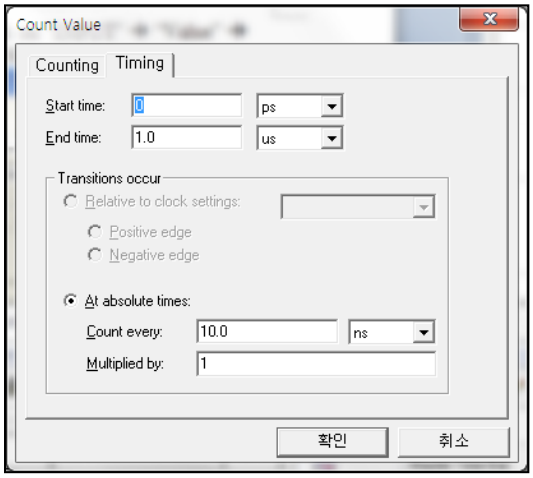


4. Click "OK". In the following pop-up, select Unsigned Decimal.

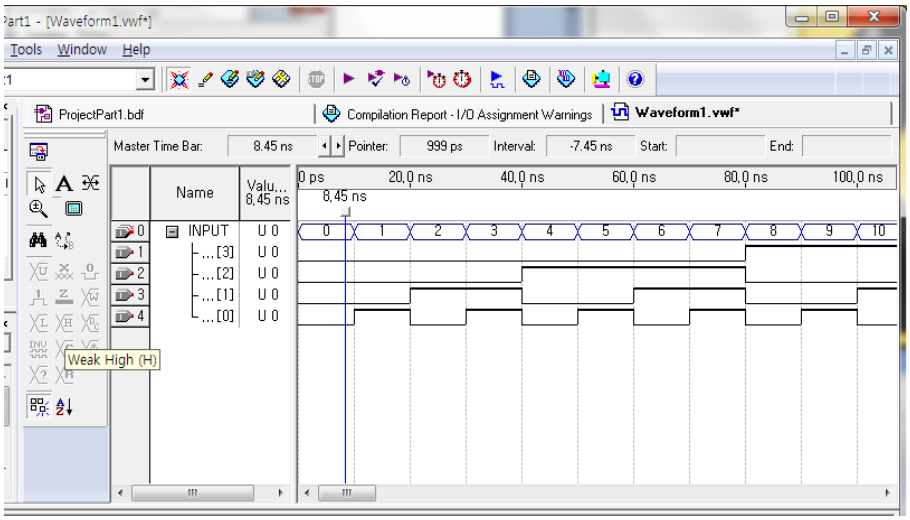


5. Now, the Vector Wave Form window shows signal "INPUT". Right click on "INPUT" → "Value" → "Count Value". The default should show the following options. Click "OK".

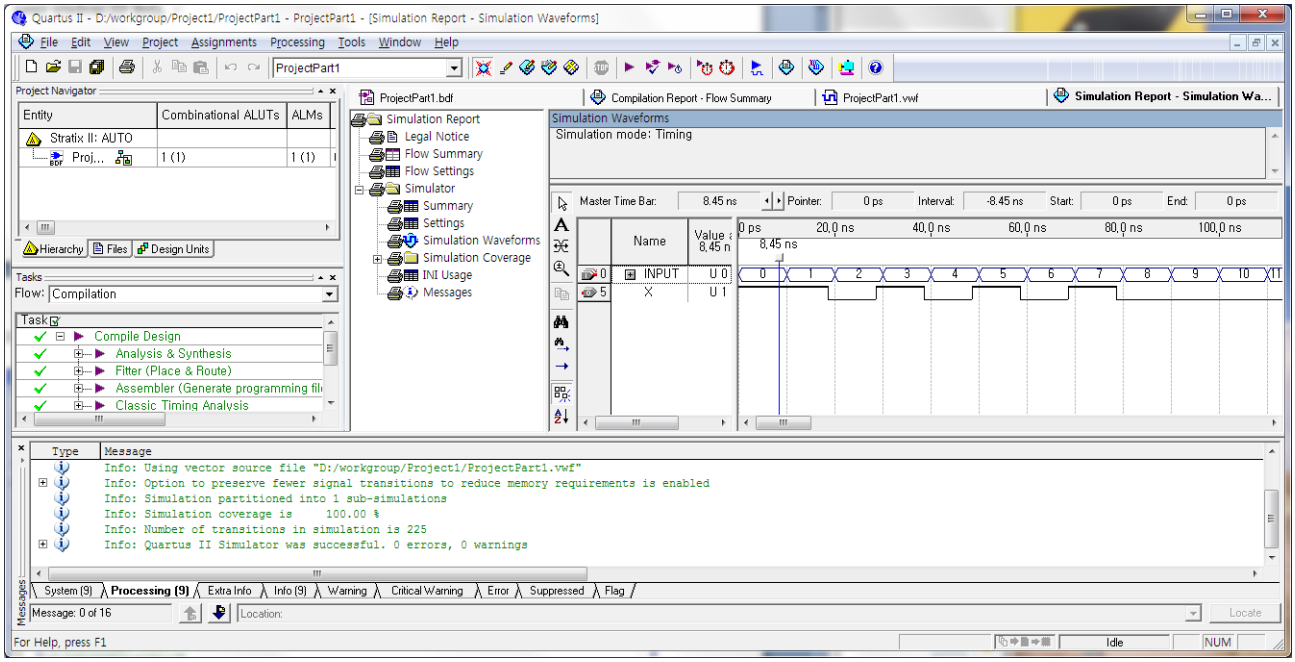




6. You can see the following Wave Form. It is your input signal, INPUT[3..0]. You can also expand INPUT[3..0] to see each individual input signal. Save the file as the same name to your project. You can zoom-in or zoom-out by pressing "Ctrl+space" or "Ctrl+Shift+space".

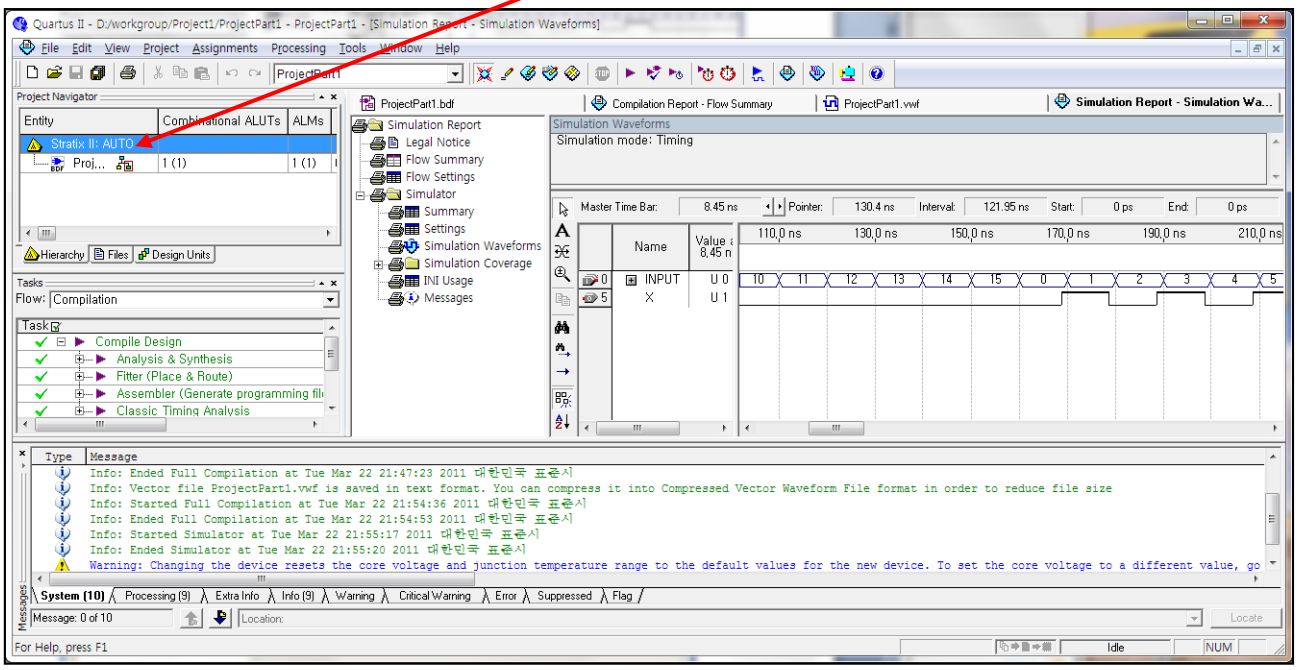


7. Now, you have your circuit implementation and have configured your input signal. Let's start simulation. Go to "Processing" → "Start Simulation". When you change your schematic, you should compiler first before starting simulation. The following diagram is the results. As expected, the output is high when the input signal is 0, 2, 4, or 6. **Now you finish all requirements for the first lab. However, it is with Chulsu's ID. So, go ahead with your own Student ID.**

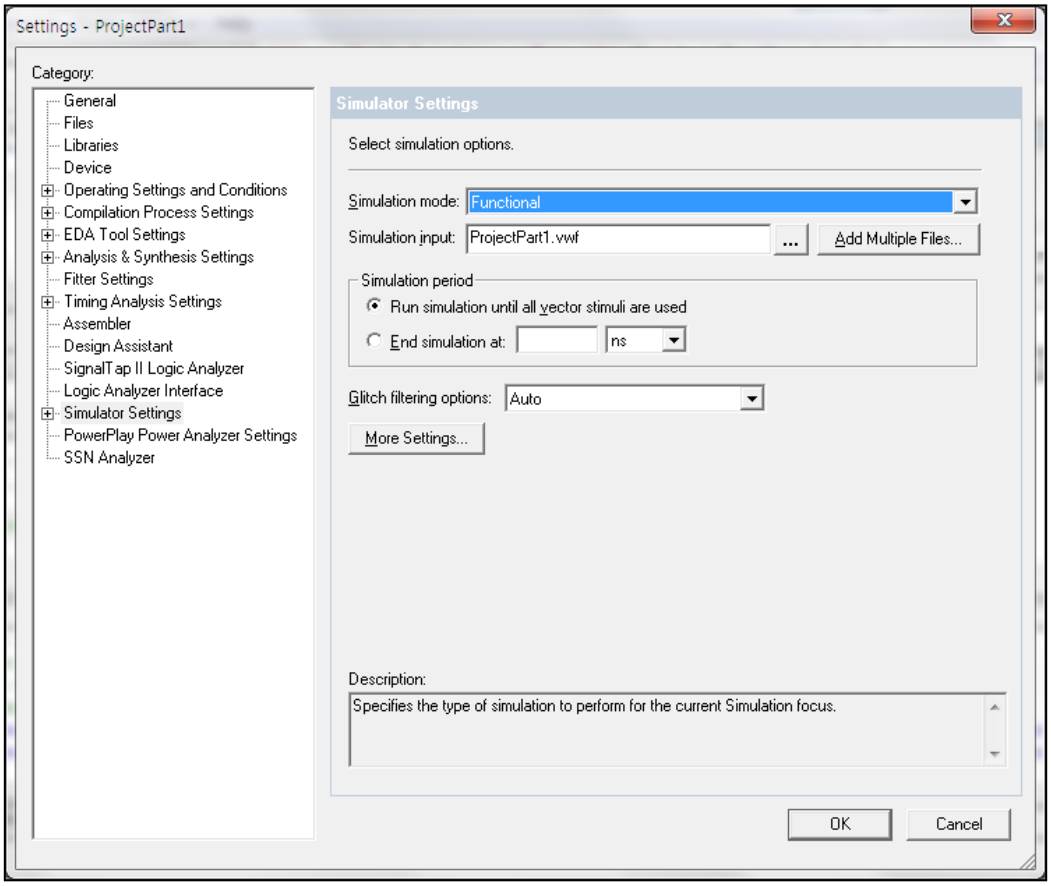


Note: we are simulating only functional output of the logic; we don't have to consider any timing delay in this lab. You can configure the simulator option as "functional" simulation mode.

First double click on the device of the Entity window. Here!



Secondly, select "Simulator Settings" → Simulation mode: Functional.



Remember: If you want to set Simulation mode as "Functional", **<u>you should generate Functional Simulation Netlist first</u>**, and then you can simulate the waveform. ("Processing" → "Generate Functional Simulation Netlist)