Switching Circuit & Logic Design

Basic Logic Design Practicing - Quartus II Schematic

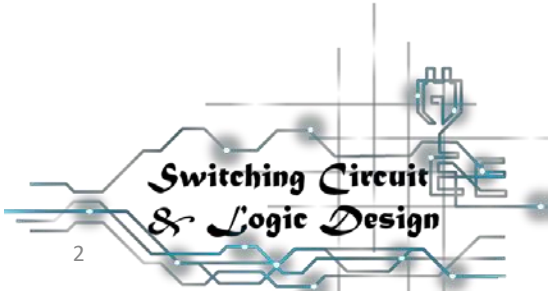
Logic Design & Simulation Practicing

--Introduction to Quartus II Schematic Tools

Lecturer: TA 蔡承佑 (BL-430)
r10943014@ntu.edu.tw

交換電路與邏輯設計課程
Professors: 吳安宇 盧奕璋 江蕙如

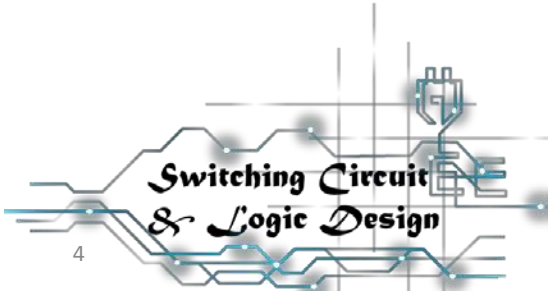
臺灣大學



Outline

Combinational Circuit

Design Flow Introduction

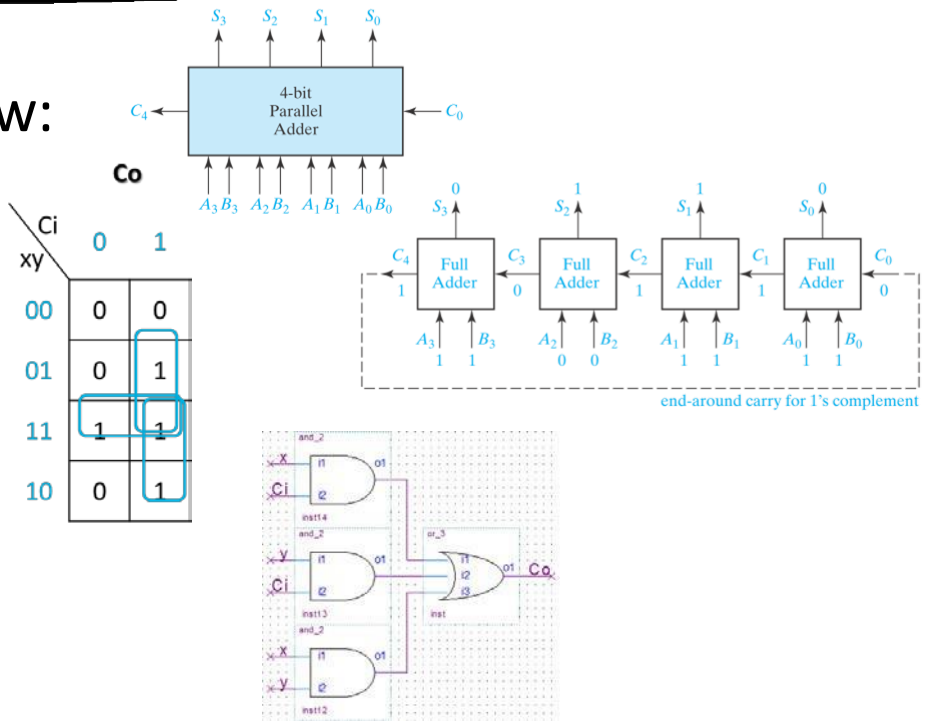


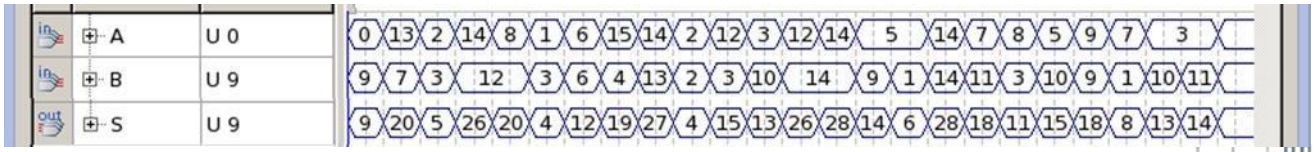
Comb. Ckt Design flow

- Gate-level schematic design flow:
  1. Divide system - Divide big design to a system of small modules
     - For each small module, do the following steps:
  2. Truth table - Think thoroughly about your topic
  3. K-map - Simplify the logic (multi-output simplification)
  4. Boolean expression - The exact form you're going to make
  5. Draw the Ckt - Be sure meet the requirements (ex. gate count)
  6. Design with tools - Implement it (.bdf or .v)
     - You can pack partial circuit as sub-module by .bsf + .bdf
  7. Verify & debug - Test some typical patterns, simulate it by wave form

Comb. Ckt Design flow

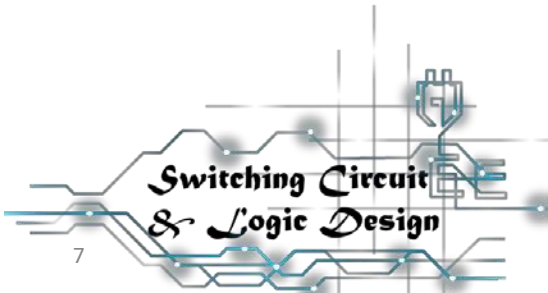
- Gate-level schematic design flow:
  1. Divide system
  2. Truth table
  3. K-map
  4. Boolean expression
  5. Draw it first
  6. Design with tools
  7. Verify & debug





Introduction to SchematicTools

Quartus II



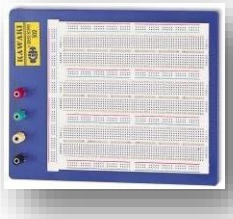
What is Quartus II ?

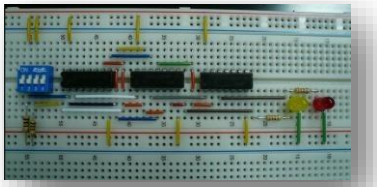
- It's a software tool for computer-aided digital circuit design
  - Proposed by Altera
  - Especially for programmable logic device (ex. FPGA) from Altera

- You can design your circuit in both schematic / HDL file on it
  - Both: Design, Simulation, Verification, Analysis and Synthesis
    - Logic (functional) simulation
    - Timing simulation
    - Vector waveform simulation (verification)
    - Logic optimization (with FPGA placement & routing)
    - Programs the logic gates on FPGA board

Steps to design & verify your circuit

1. Launch Quartus II software
   – Install it your self or run on workstation
2. Build a Quartus project
3. Build a block diagram file (bdf) and realize your circuit
   – Remember to import the logic elements file
4. Compile your design
5. Import a wave form file (vwf) (or build it yourself)
6. Run functional simulation
   – check whether your design is correct or not
   – If not, back to step #3





Tutorial for those who does not work on workstation

Install Quartus II

Install

- To avoid compatibility problem

- Virtual Machine File (.vdi) is provided

  -Window 7 + Quartus

Download Virtual Box

前往 https://www.virtualbox.org/ 下載安裝

Import .vdi file

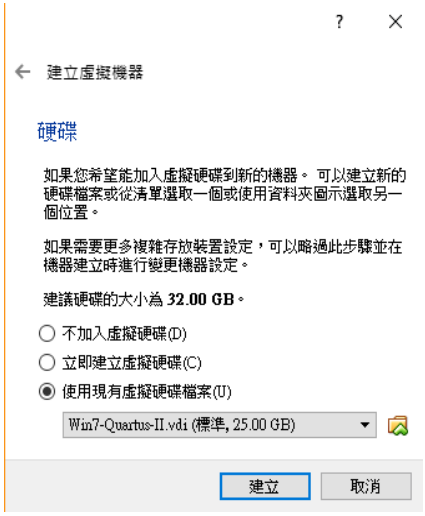
Step 1: 打開Virtual Box後，點擊新增

Step 3: 選擇記憶體大小

Step 5：點擊開啟

Step2：版本選擇Window7(32-bit)

Step4：匯入預先下載的.vdi 檔

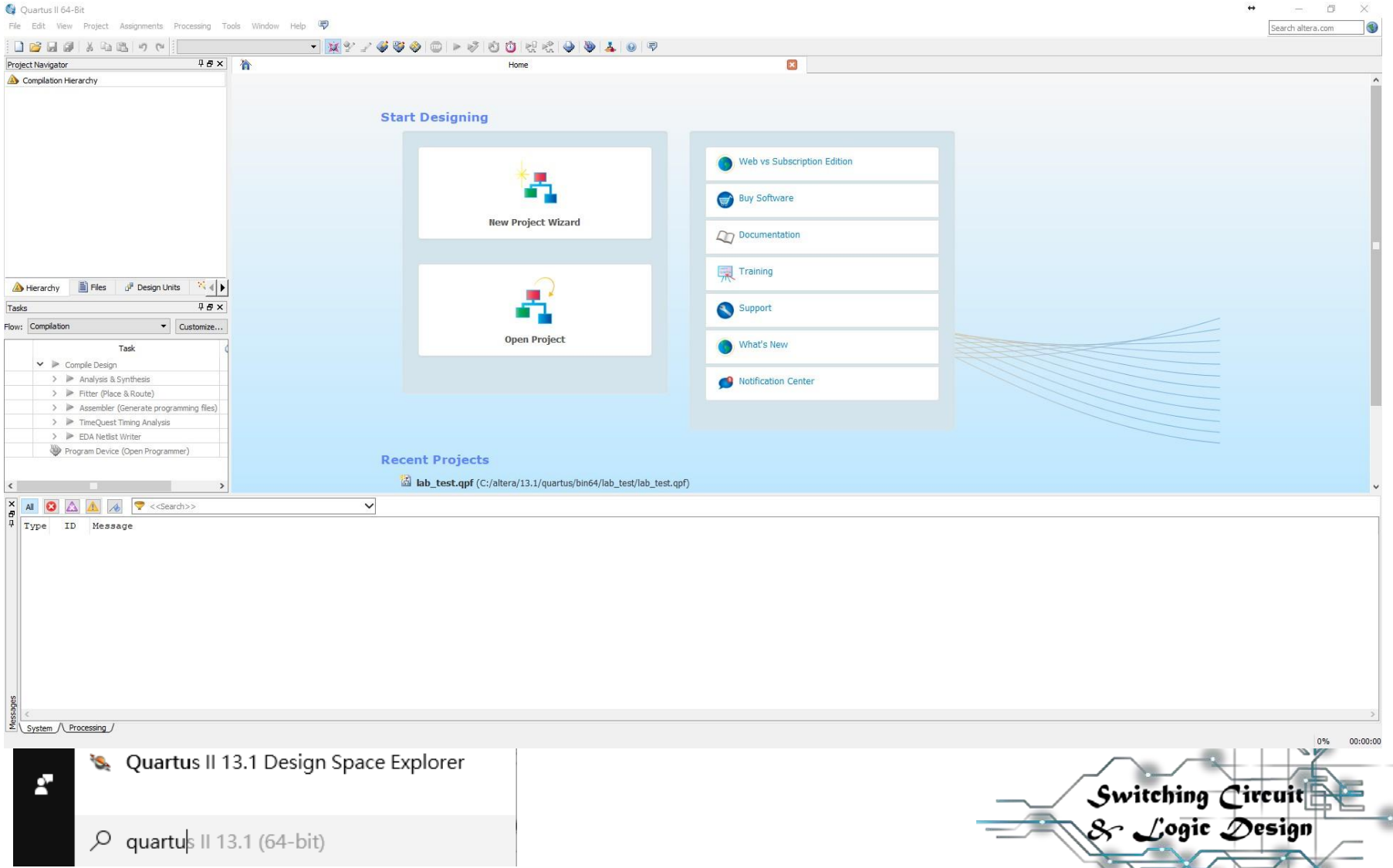


Step6：開始使用

Very first step to get in the world of Quartus II

Launch Quartus II

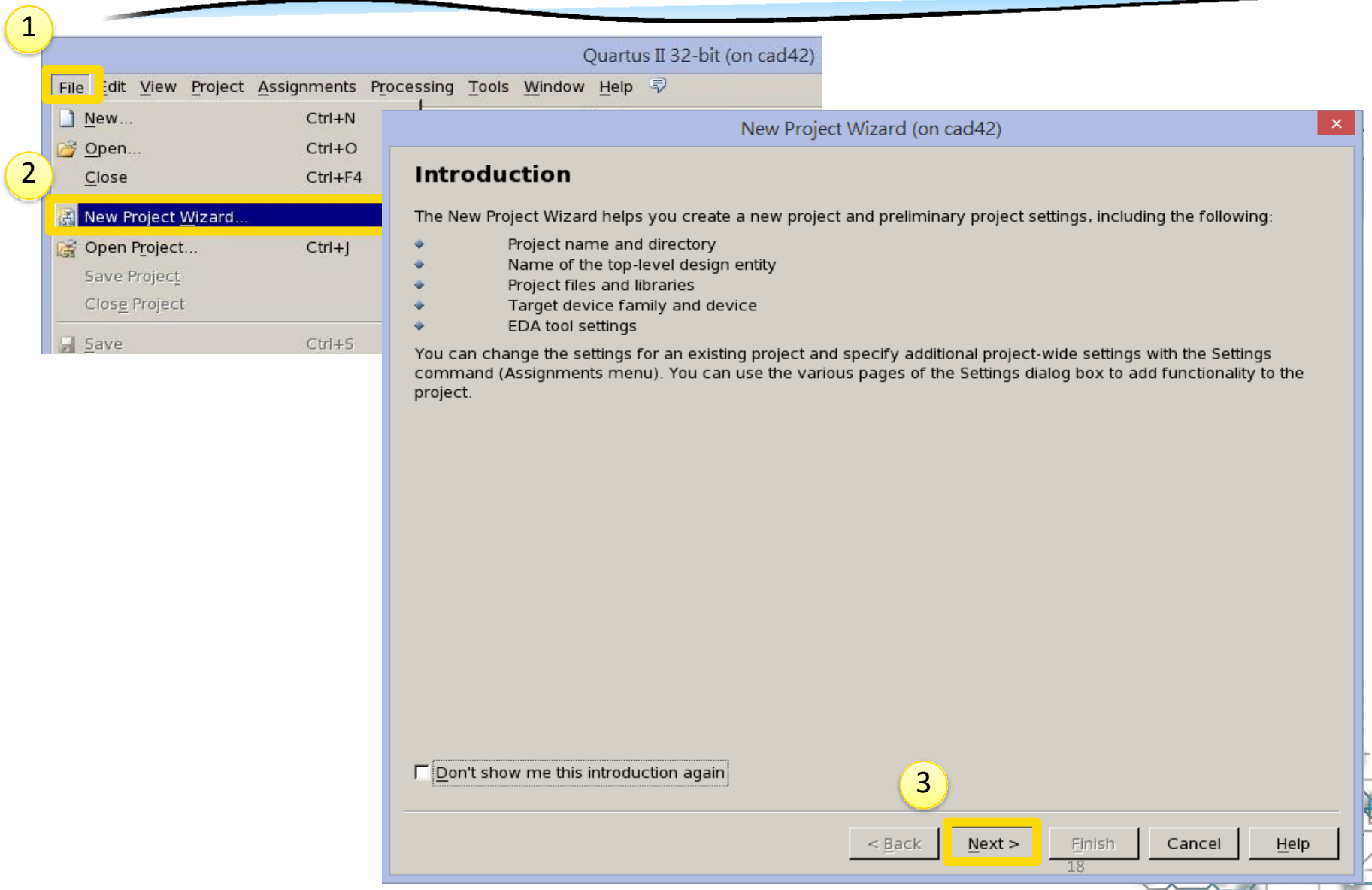
Launch Quartus II

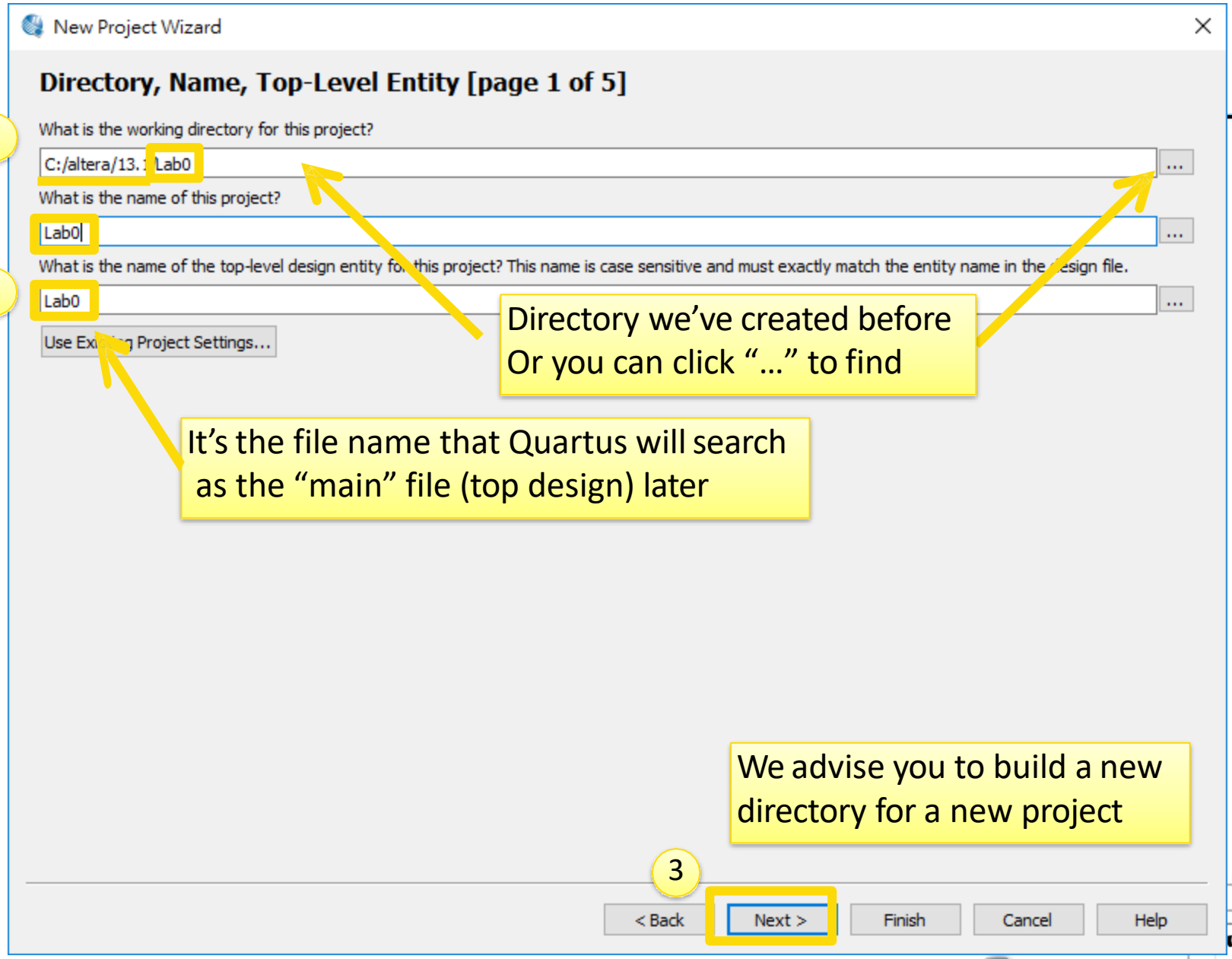


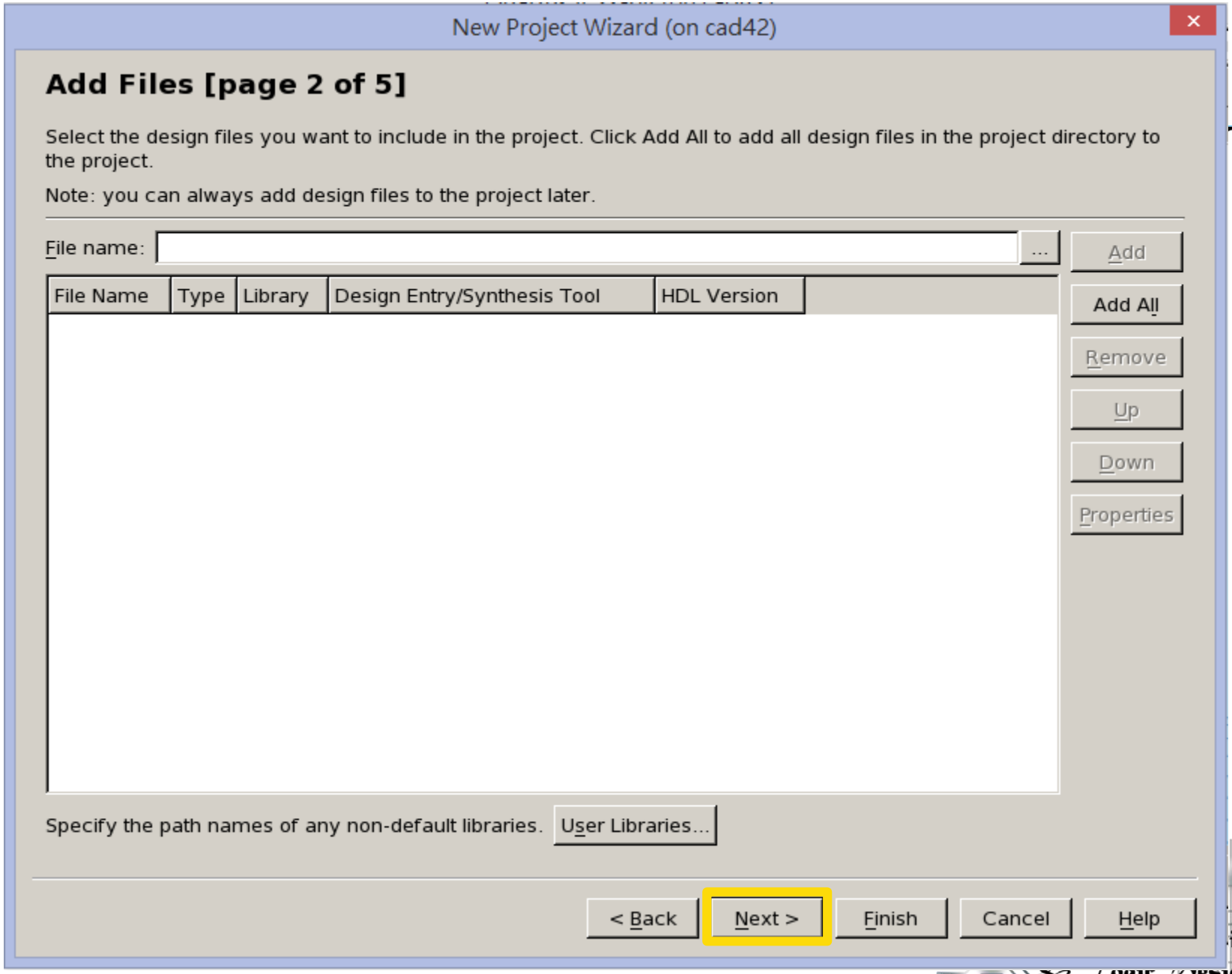
Lab0: Get it familiar!

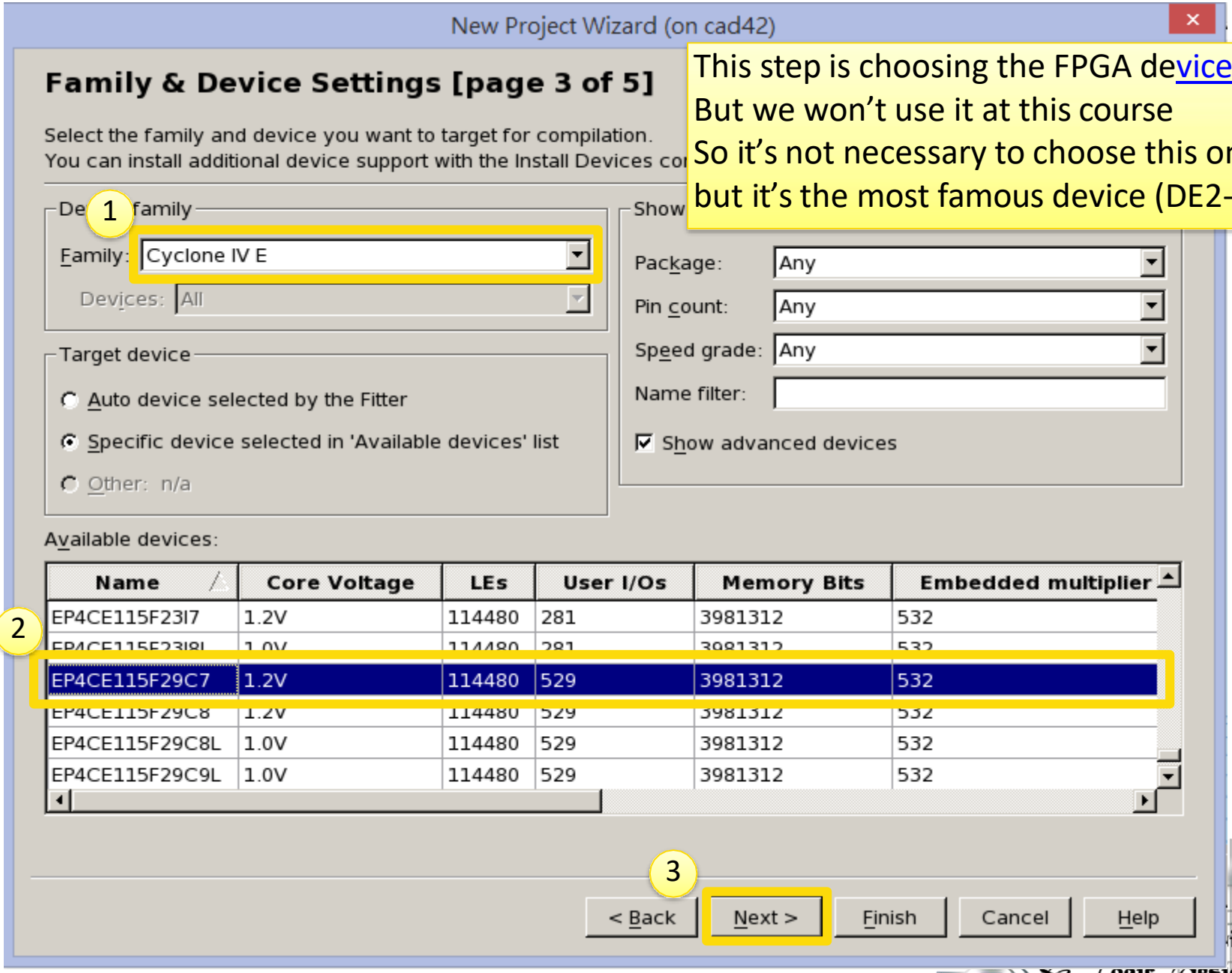
Quartus Schematic Design Trial

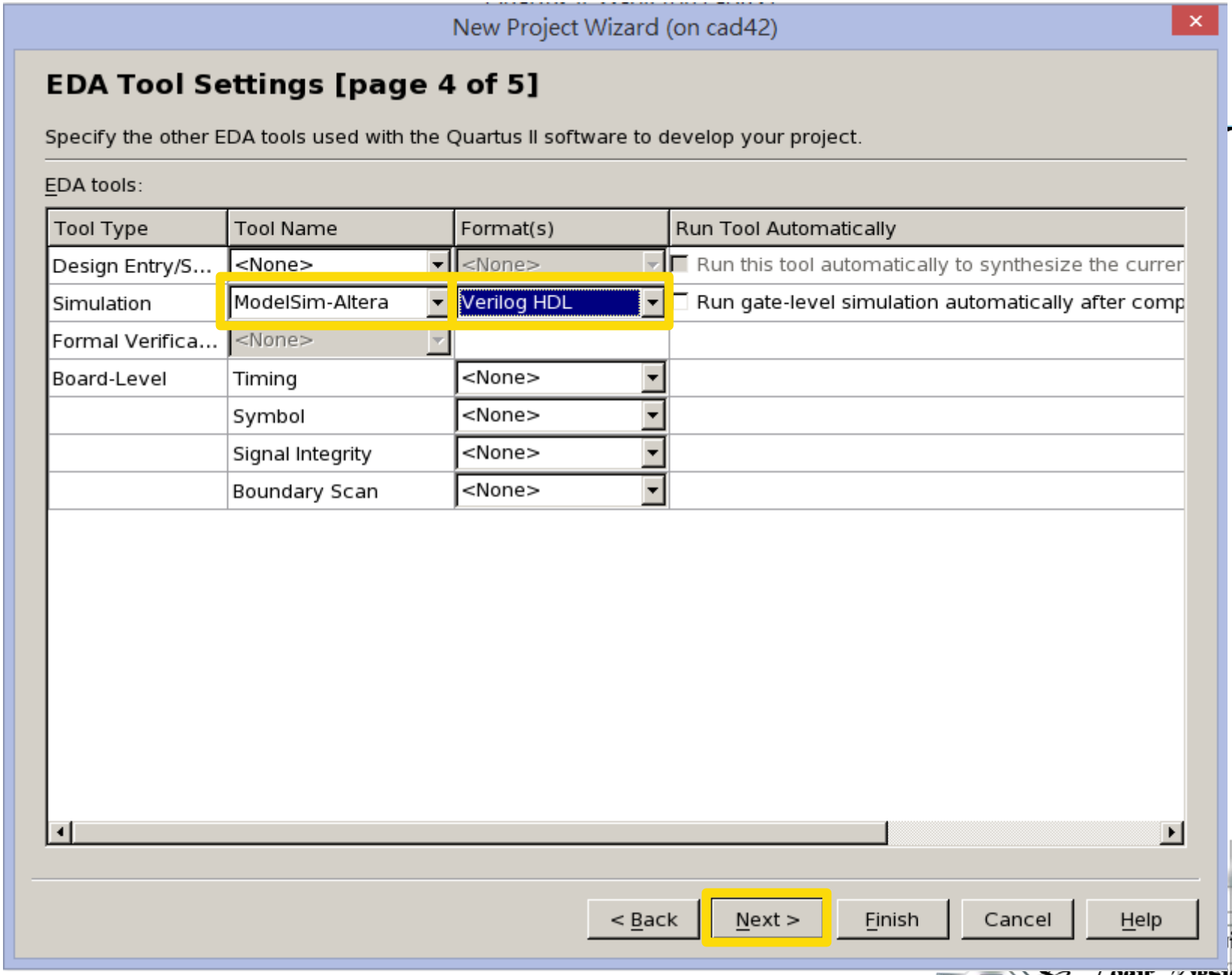
Create a New Project

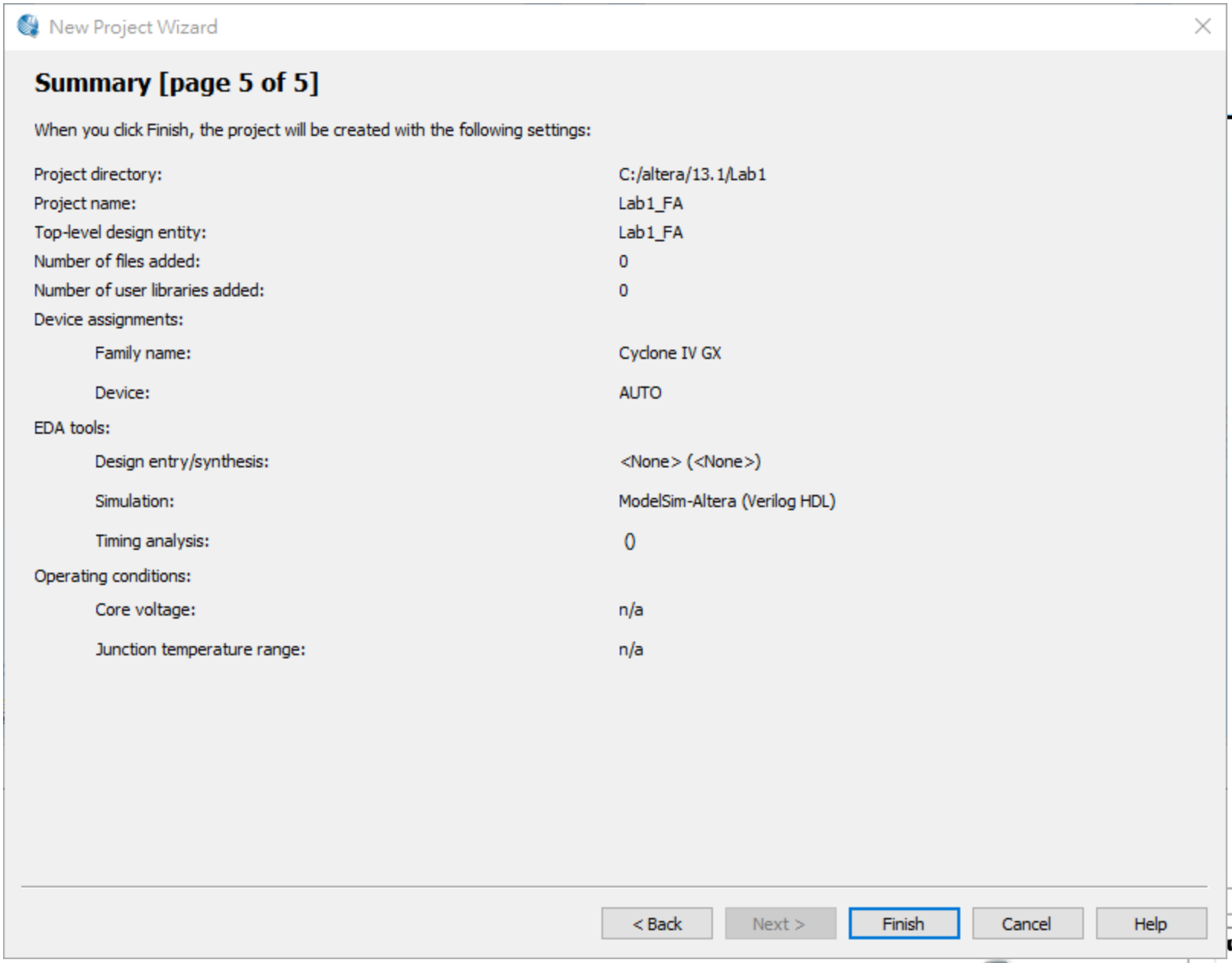



New Project Wizard
Directory, Name, Top-Level Entity [page 1 of 5]
1
What is the working directory for this project?
C:/altera/13.1 Lab0
What is the name of this project?
Lab0
What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.
2
Lab0
Use Existing Project Settings...
Directory we've created before
Or you can click "..." to find
It's the file name that Quartus will search as the "main" file (top design) later
We advise you to build a new directory for a new project
3
< Back
Next >
Finish
Cancel
Help

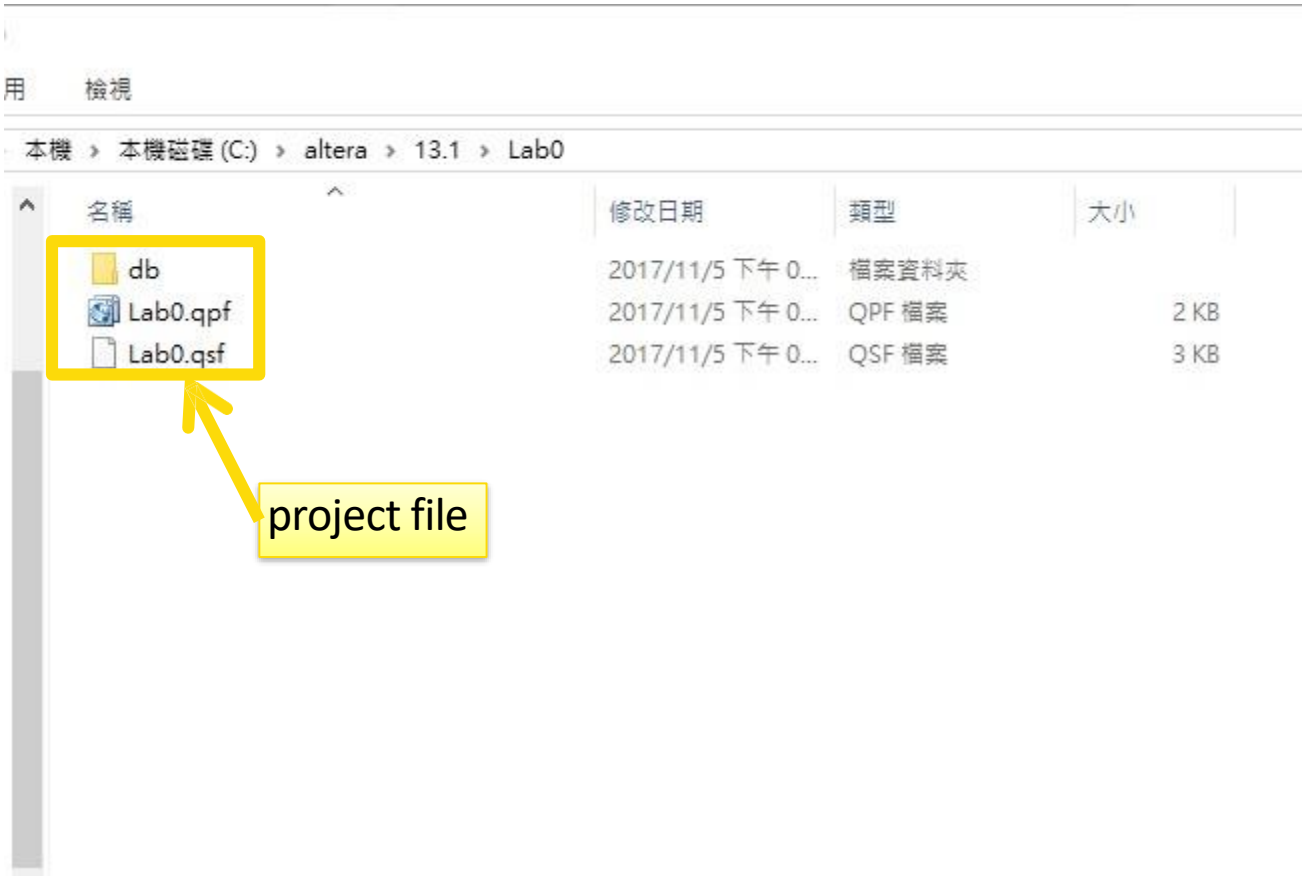

New Project Wizard (on cad42)
Add Files [page 2 of 5]
Select the design files you want to include in the project. Click Add All to add all design files in the project directory to the project.
Note: you can always add design files to the project later.
File name:
...
Add
Add All
Remove
Up
Down
Properties
File Name
Type
Library
Design Entry/Synthesis Tool
HDL Version
Specify the path names of any non-default libraries.
User Libraries...
< Back
Next >
Finish
Cancel
Help


New Project Wizard (on cad42)
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 co
This step is choosing the FPGA device
But we won't use it at this course
So it's not necessary to choose this one,
but it's the most famous device (DE2-115)
1
Family:
Cyclone IV E
Devices:
All
Target device
Auto device selected by the Fitter
Specific device selected in 'Available devices' list
Other: n/a
Package:
Any
Pin count:
Any
Speed grade:
Any
Name filter:
Show advanced devices
Available devices:
Name
Core Voltage
LEs
User I/Os
Memory Bits
Embedded multiplier
2
EP4CE115F23I7 1.2V 114480 281 3981312 532
EP4CE115F29C7 1.2V 114480 529 3981312 532
EP4CE115F29C8L 1.0V 114480 529 3981312 532
EP4CE115F29C9L 1.0V 114480 529 3981312 532
3
< Back
Next >
Finish
Cancel
Help

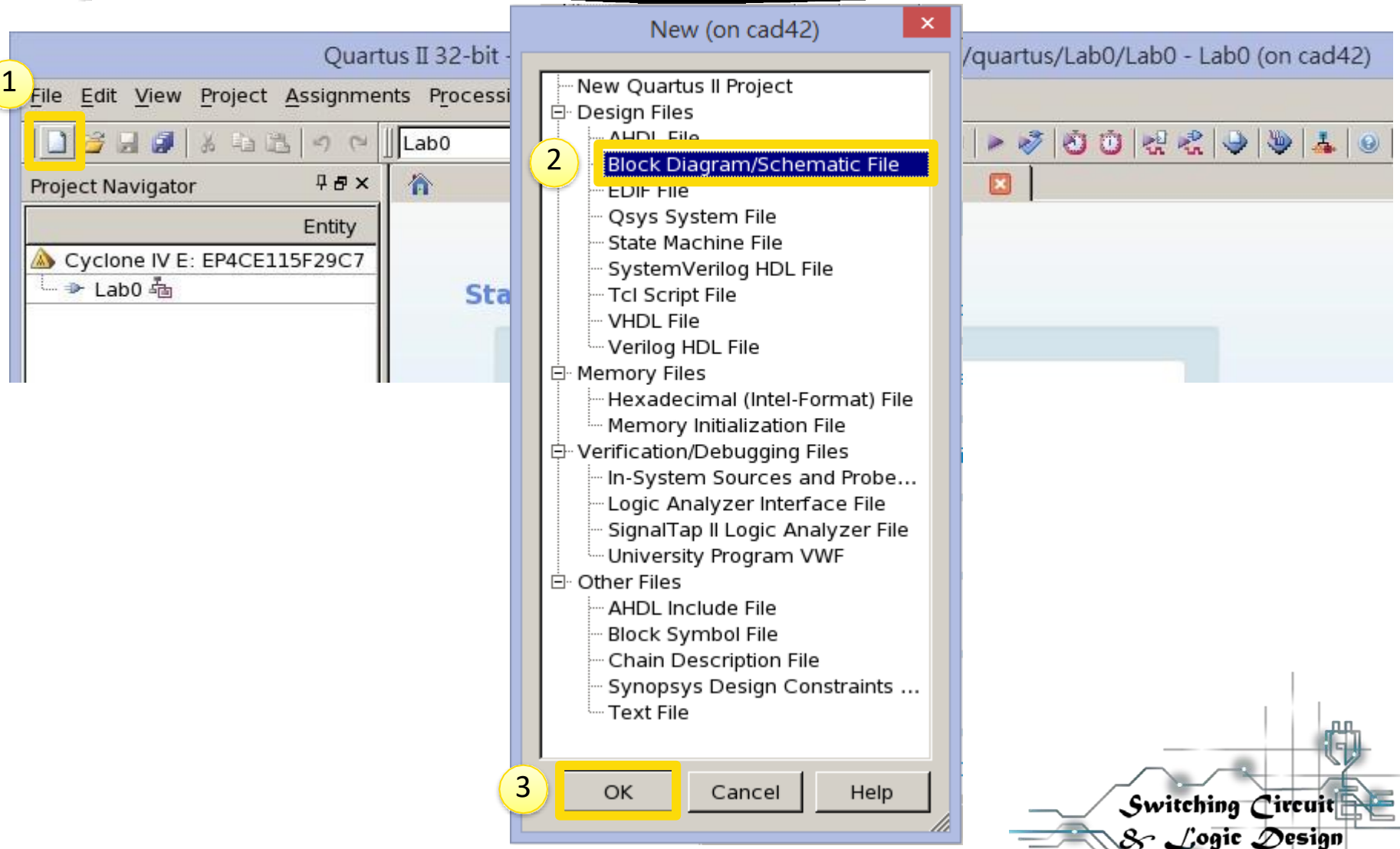

New Project Wizard (on cad42)
EDA Tool Settings [page 4 of 5]
Specify the other EDA tools used with the Quartus II software to develop your project.
EDA tools:
Tool Type
Tool Name
Format(s)
Run Tool Automatically
Design Entry/S...
<None>
<None>
Run this tool automatically to synthesize the currer
Simulation
ModelSim-Altera
Verilog HDL
Run gate-level simulation automatically after comp
Formal Verifica...
<None>
Board-Level
Timing
<None>
Symbol
<None>
Signal Integrity
<None>
Boundary Scan
<None>
< Back
Next >
Finish
Cancel
Help


New Project Wizard
Summary [page 5 of 5]
When you click Finish, the project will be created with the following settings:
Project directory: C:/altera/13.1/Lab1
Project name: Lab1_FA
Top-level design entity: Lab1_FA
Number of files added: 0
Number of user libraries added: 0
Device assignments:
Family name: Cyclone IV GX
Device: AUTO
EDA tools:
Design entry/synthesis: <None> (<None>)
Simulation: ModelSim-Altera (Verilog HDL)
Timing analysis: 0
Operating conditions:
Core voltage: n/a
Junction temperature range: n/a
< Back
Next >
Finish
Cancel
Help

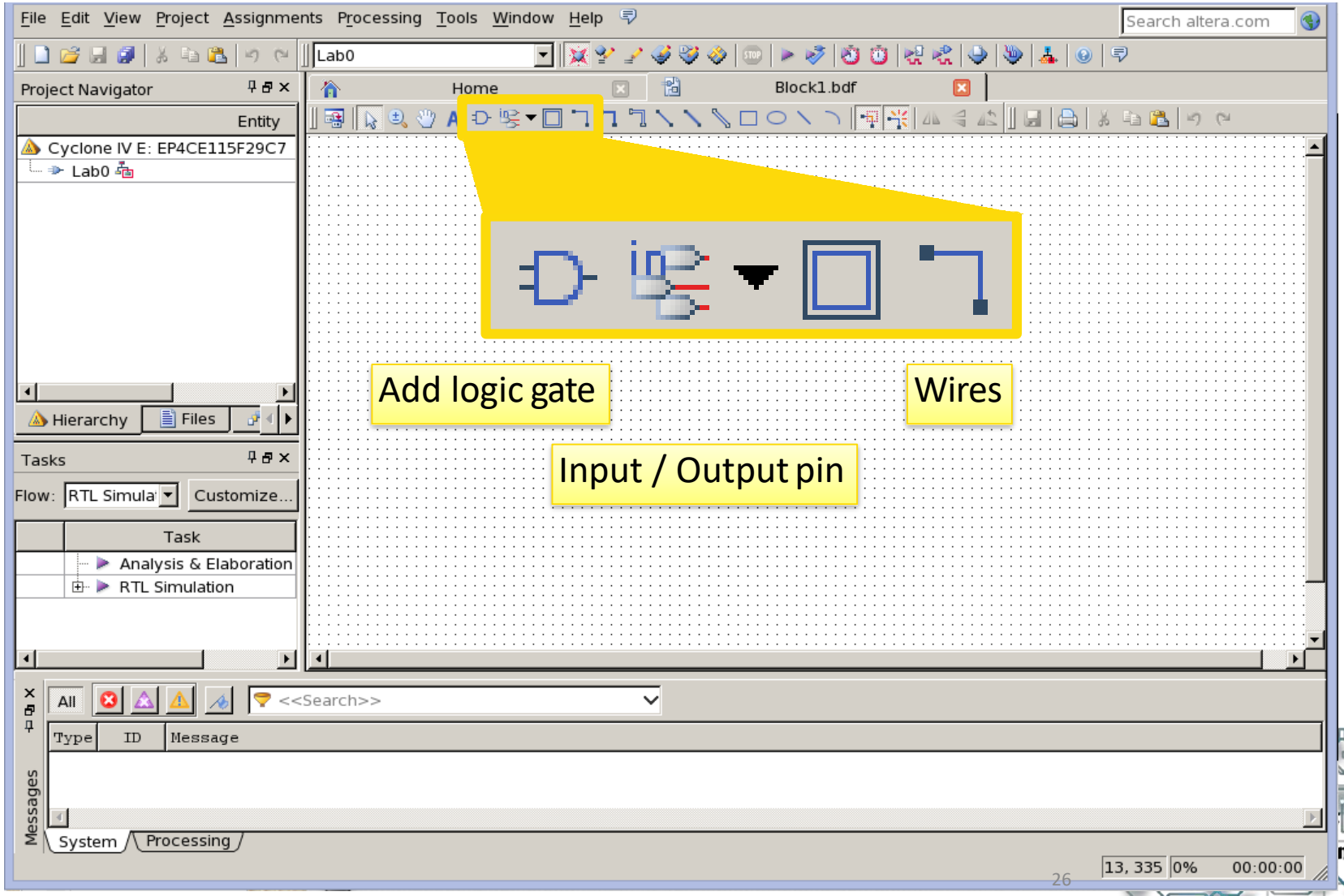
Create a New Project



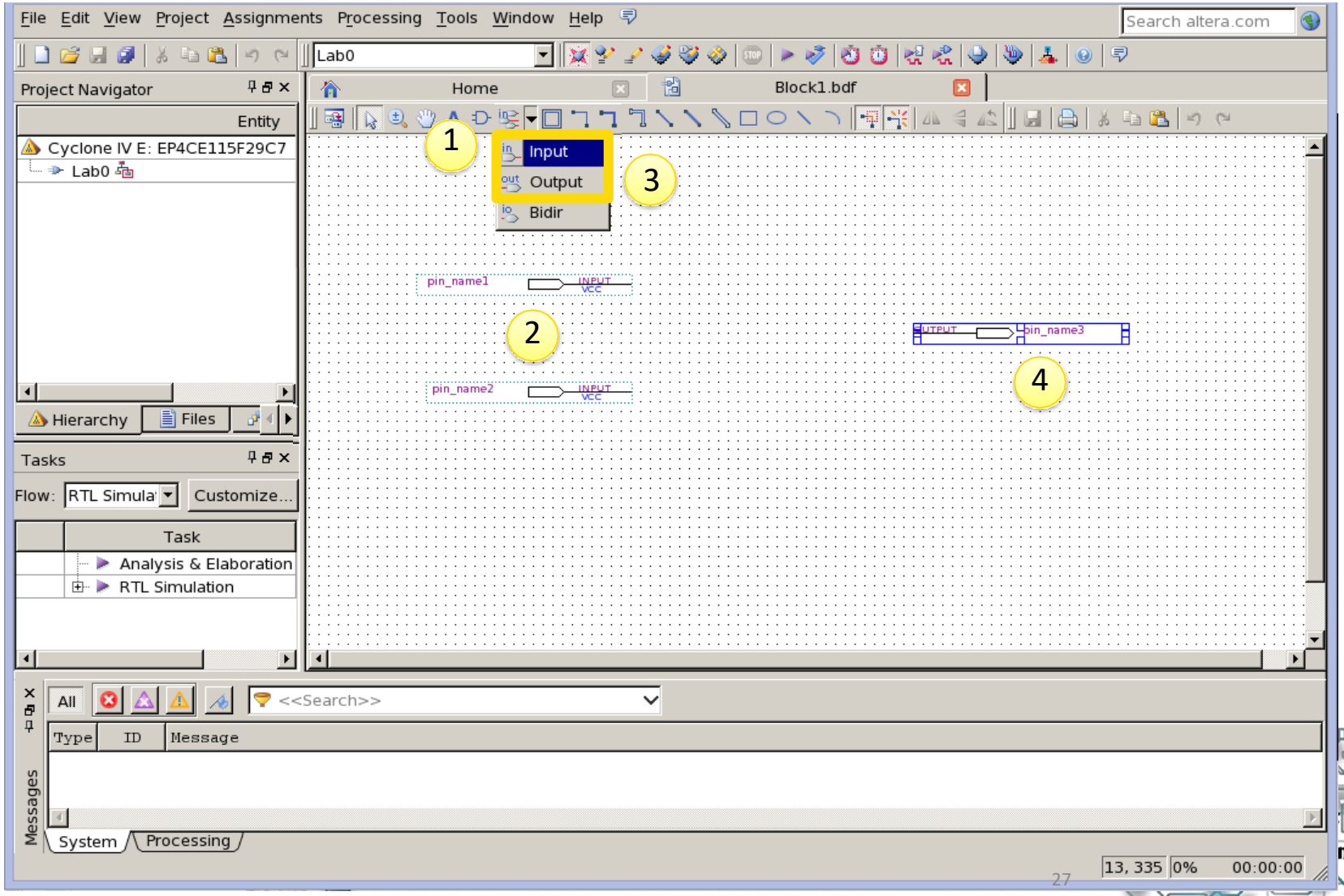
Create a Block Diagram File



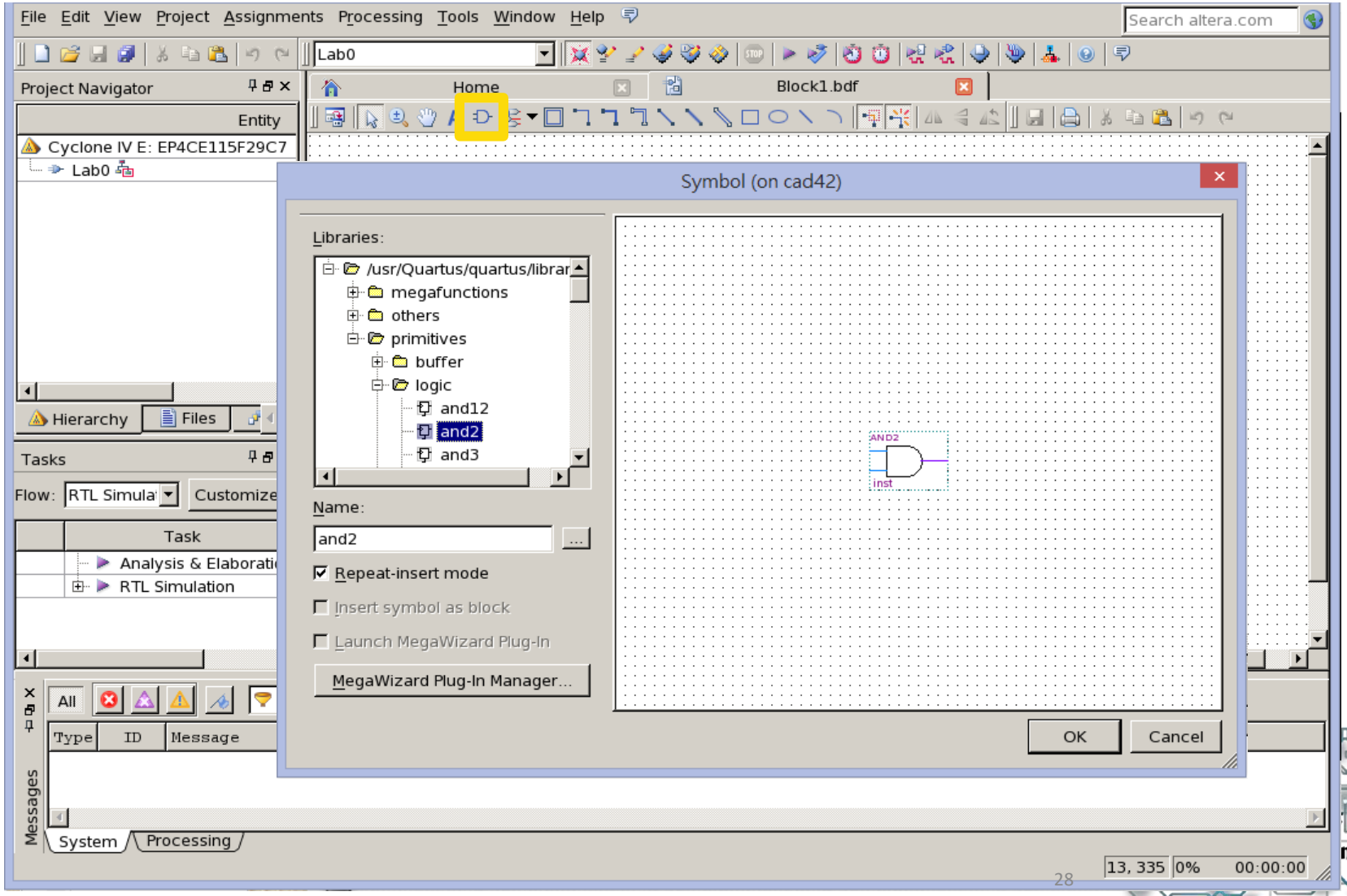
Create a Block Diagram File



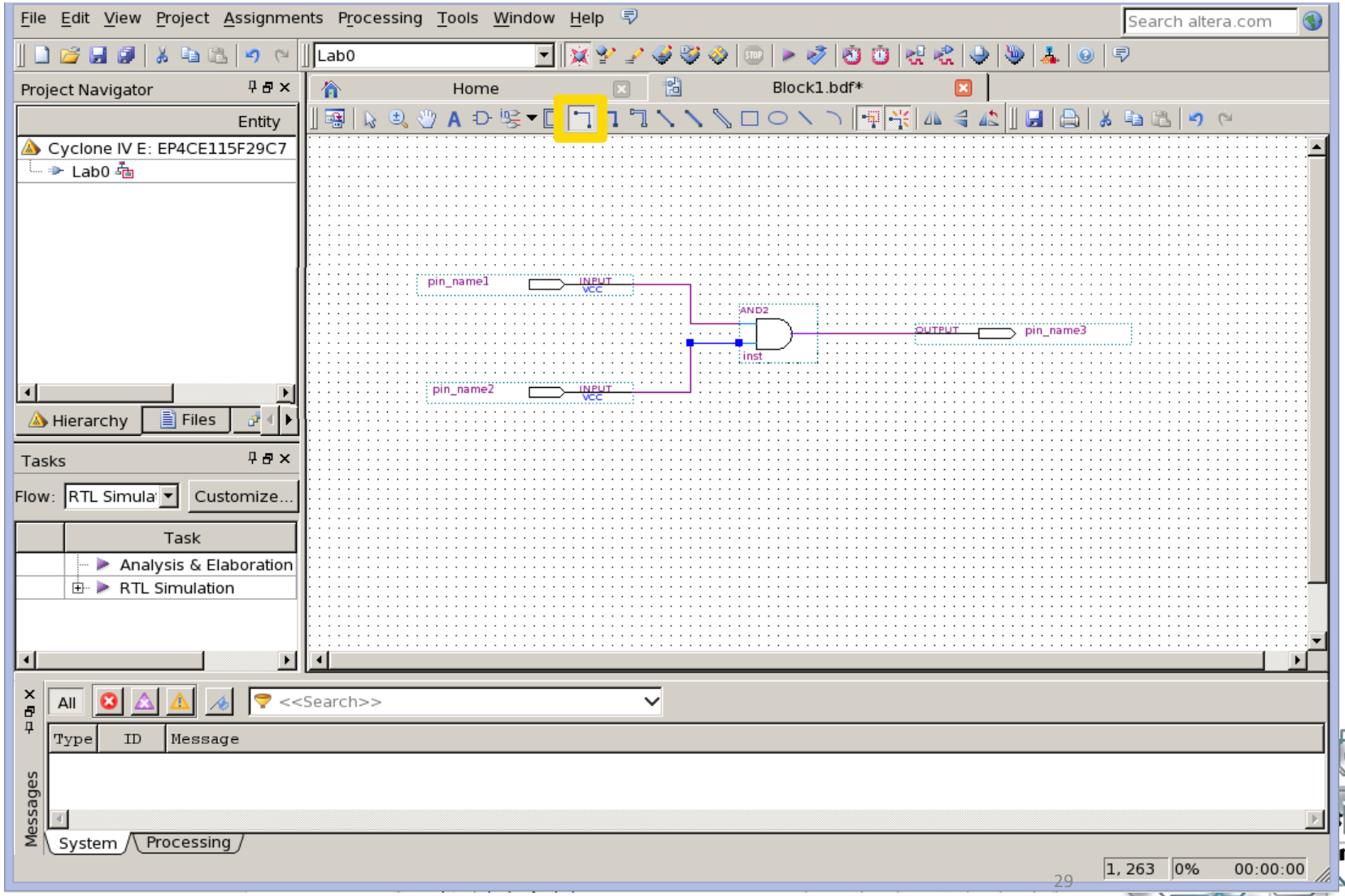
Example


File Edit View Project Assignments Processing Tools Window Help
Search altera.com
Lab0
Project Navigator
Entity
Cyclone IV E: EP4CE115F29C7
Lab0
Hierarchy
Files
Tasks
Flow: RTL Simula
Customize...
Task
Analysis & Elaboration
RTL Simulation
Home
Block1.bdf
Input
Output
Bidir
1
2
3
4
pin_name1
INPUT
VCC
pin_name2
INPUT
VCC
OUTPUT
pin_name3
All
<<Search>>
Type
ID
Message
Messages
System
Processing
13, 335
0%
00:00:00

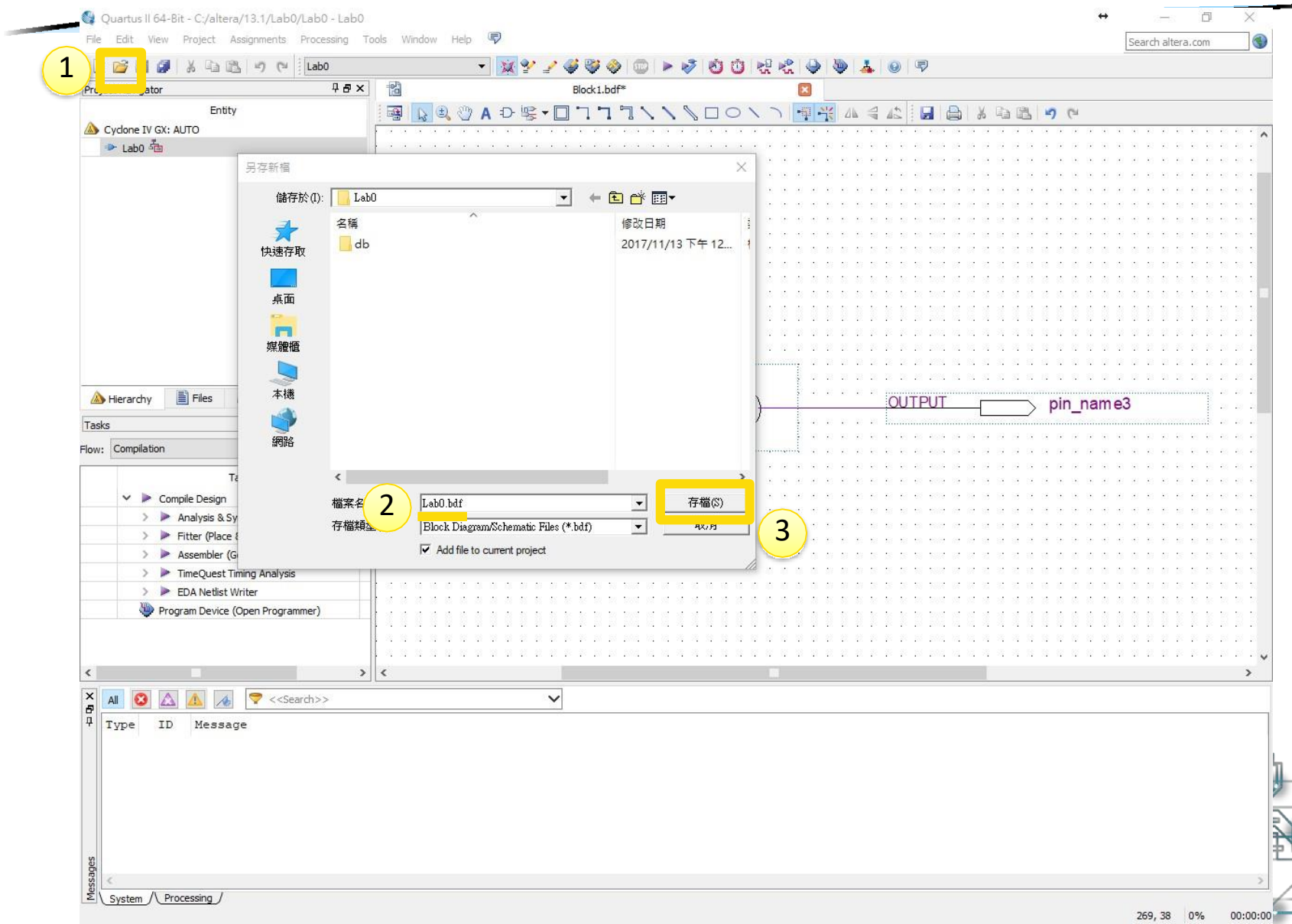
Example


File Edit View Project Assignments Processing Tools Window Help
Search altera.com
Lab0
Project Navigator
Entity
Cyclone IV E: EP4CE115F29C7
Lab0
Hierarchy
Files
Tasks
Flow: RTL Simula
Customize
Task
Analysis & Elaborati
RTL Simulation
Home
Block1.bdf
Symbol (on cad42)
Libraries:
/usr/Quartus/quartus/librar
megafunctions
others
primitives
buffer
logic
and12
and2
and3
Name:
and2
Repeat-insert mode
Insert symbol as block
Launch MegaWizard Plug-In
MegaWizard Plug-In Manager...
AND2
inst
OK
Cancel
All
Type
ID
Message
Messages
System
Processing
13, 335
0%
00:00:00

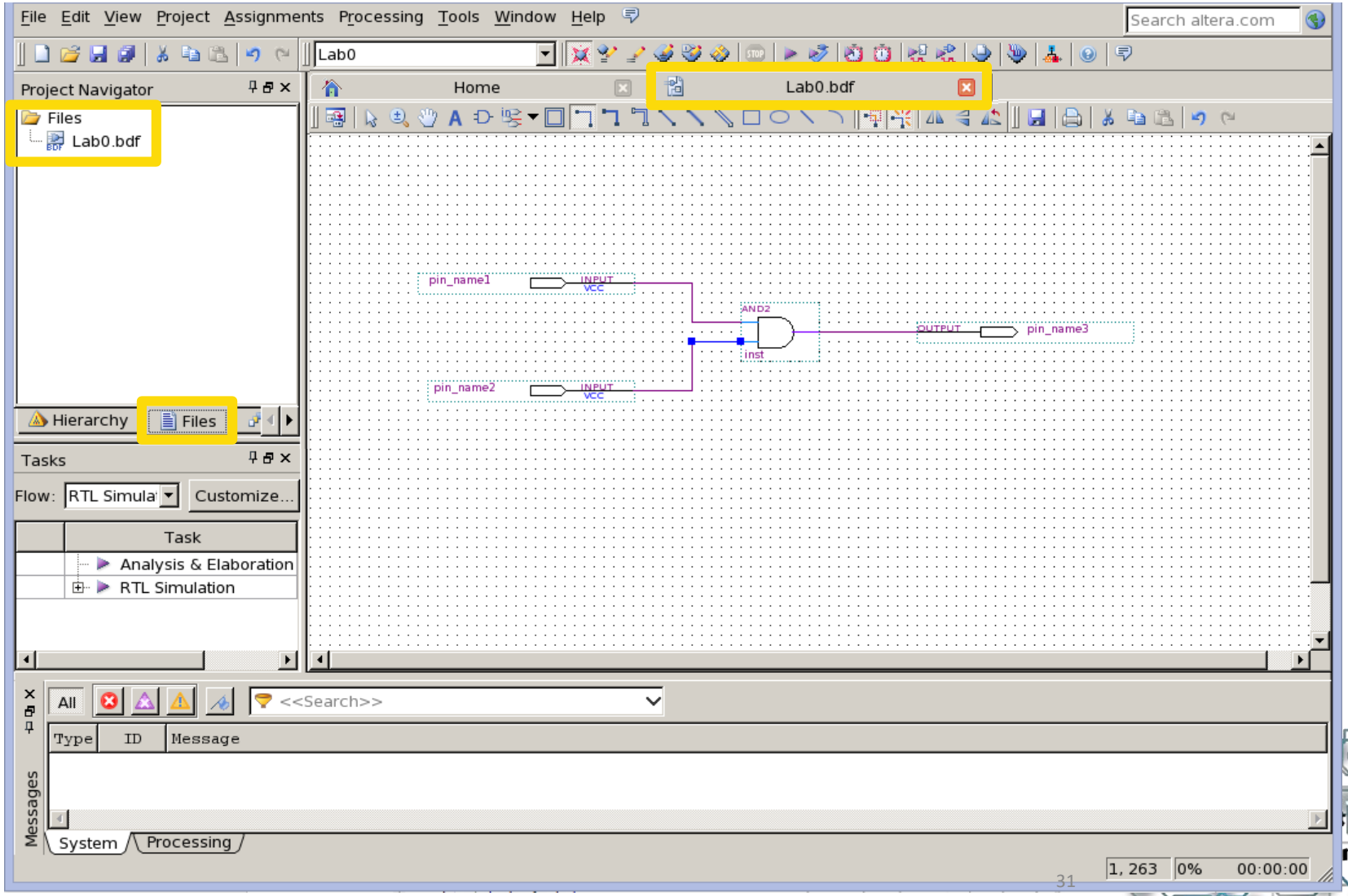
Example


File Edit View Project Assignments Processing Tools Window Help
Search altera.com
Lab0
Project Navigator
Entity
Cyclone IV E: EP4CE115F29C7
Lab0
Hierarchy
Files
Home
Block1.bdf*
pin_name1
INPUT
VCC
pin_name2
INPUT
VCC
AND2
inst
OUTPUT
pin_name3
Tasks
Flow: RTL Simula
Customize...
Task
Analysis & Elaboration
RTL Simulation
All
<<Search>>
Type
ID
Message
Messages
System
Processing
1, 263
0%
00:00:00

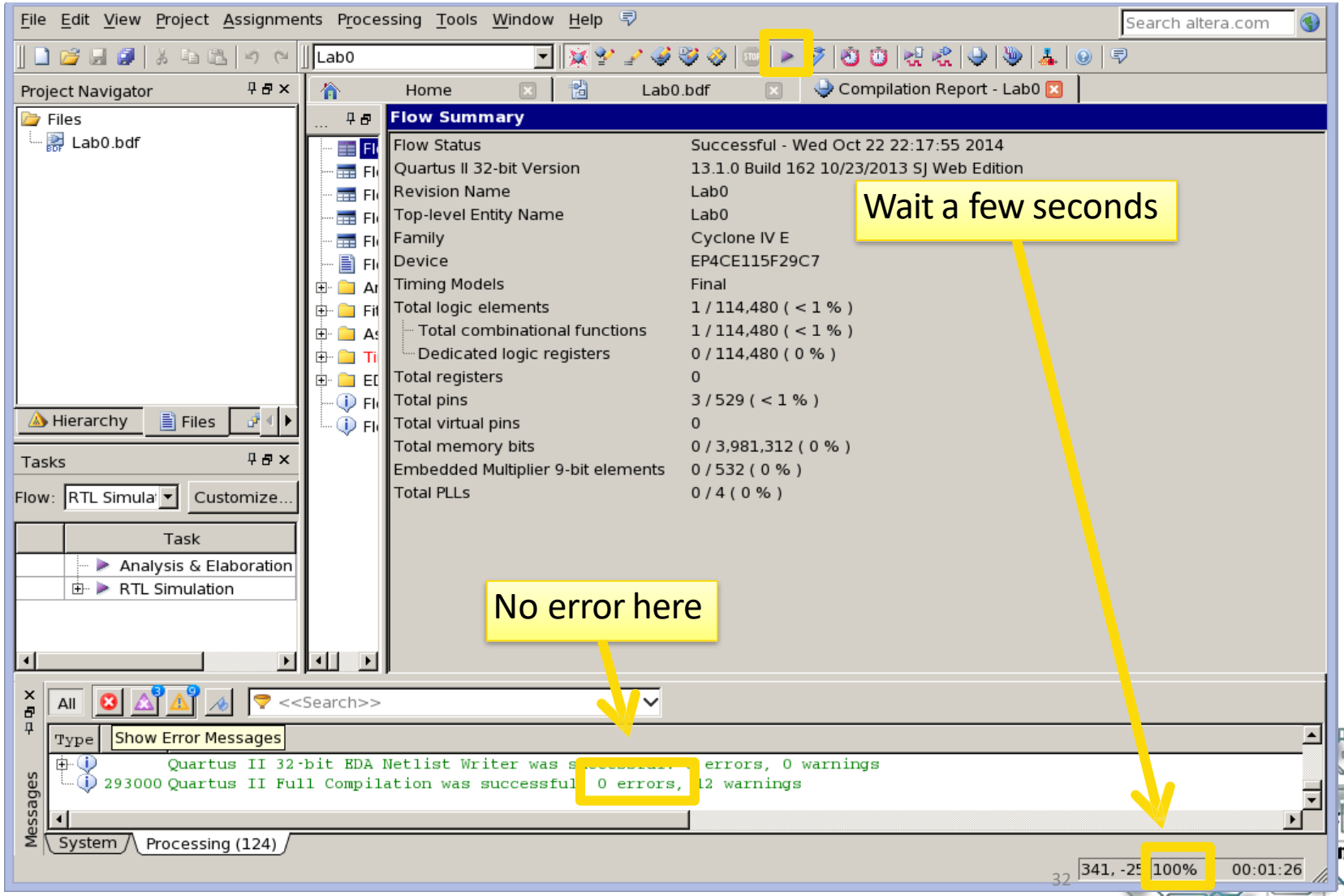
Save it


1
2
3
Lab0.bdf
存檔(S)
Add file to current project
OUTPUT
pin_name3

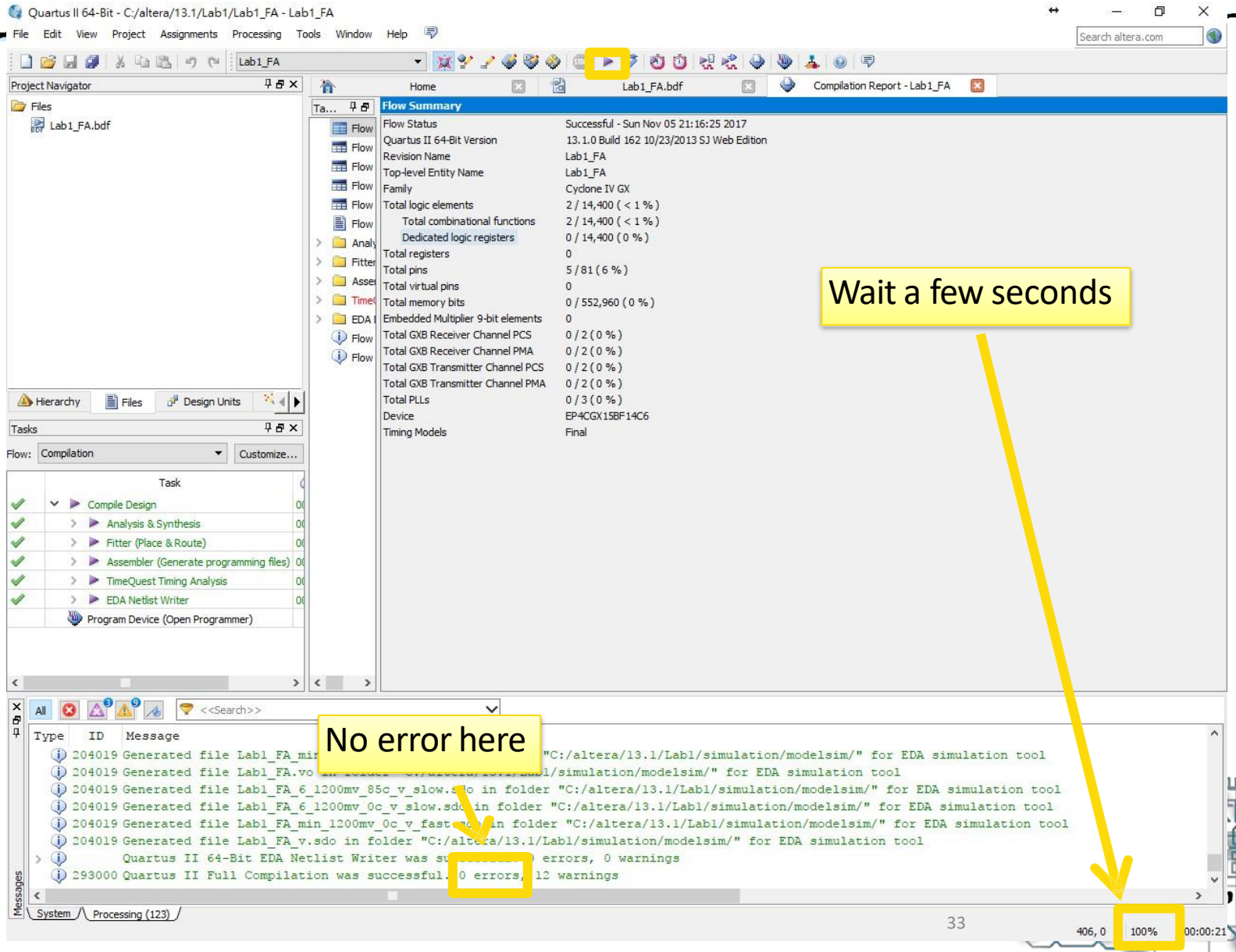
Save it


File Edit View Project Assignments Processing Tools Window Help
Search altera.com
Lab0
Project Navigator
Files
Lab0.bdf
Home
Lab0.bdf
pin_name1
INPUT
VCC
AND2
OUTPUT
pin_name3
inst
pin_name2
INPUT
VCC
Hierarchy
Files
Tasks
Flow: RTL Simula
Customize...
Task
Analysis & Elaboration
RTL Simulation
All
<<Search>>
Type
ID
Message
Messages
System
Processing
1, 263
0%
00:00:00

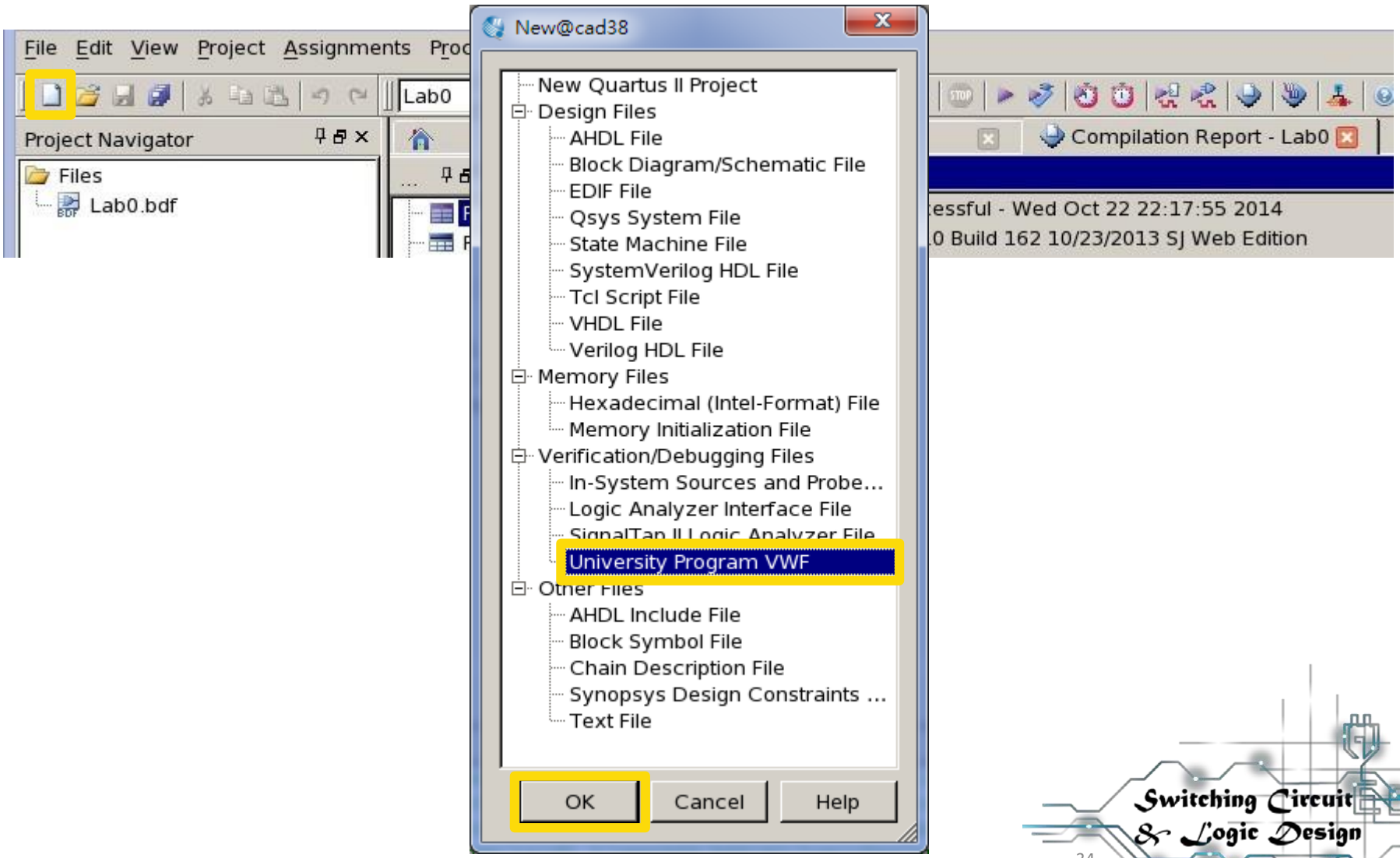
Compile


File Edit View Project Assignments Processing Tools Window Help
Search altera.com
Lab0
Project Navigator
Files
Lab0.bdf
Hierarchy
Files
Tasks
Flow: RTL Simula
Customize...
Task
Analysis & Elaboration
RTL Simulation
Home
Lab0.bdf
Compilation Report - Lab0
Flow Summary
Flow Status Successful - Wed Oct 22 22:17:55 2014
Quartus II 32-bit Version 13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name Lab0
Top-level Entity Name Lab0
Family Cyclone IV E
Device EP4CE115F29C7
Timing Models Final
Total logic elements 1 / 114,480 (< 1 %)
Total combinational functions 1 / 114,480 (< 1 %)
Dedicated logic registers 0 / 114,480 (0 %)
Total registers 0
Total pins 3 / 529 (< 1 %)
Total virtual pins 0
Total memory bits 0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements 0 / 532 (0 %)
Total PLLs 0 / 4 (0 %)
Wait a few seconds
No error here
All
<<Search>>
Type
Show Error Messages
Quartus II 32-bit EDA Netlist Writer was s errors, 0 warnings
293000 Quartus II Full Compilation was successful 0 errors, 12 warnings
Messages
System
Processing (124)
341, -25
100%
00:01:26

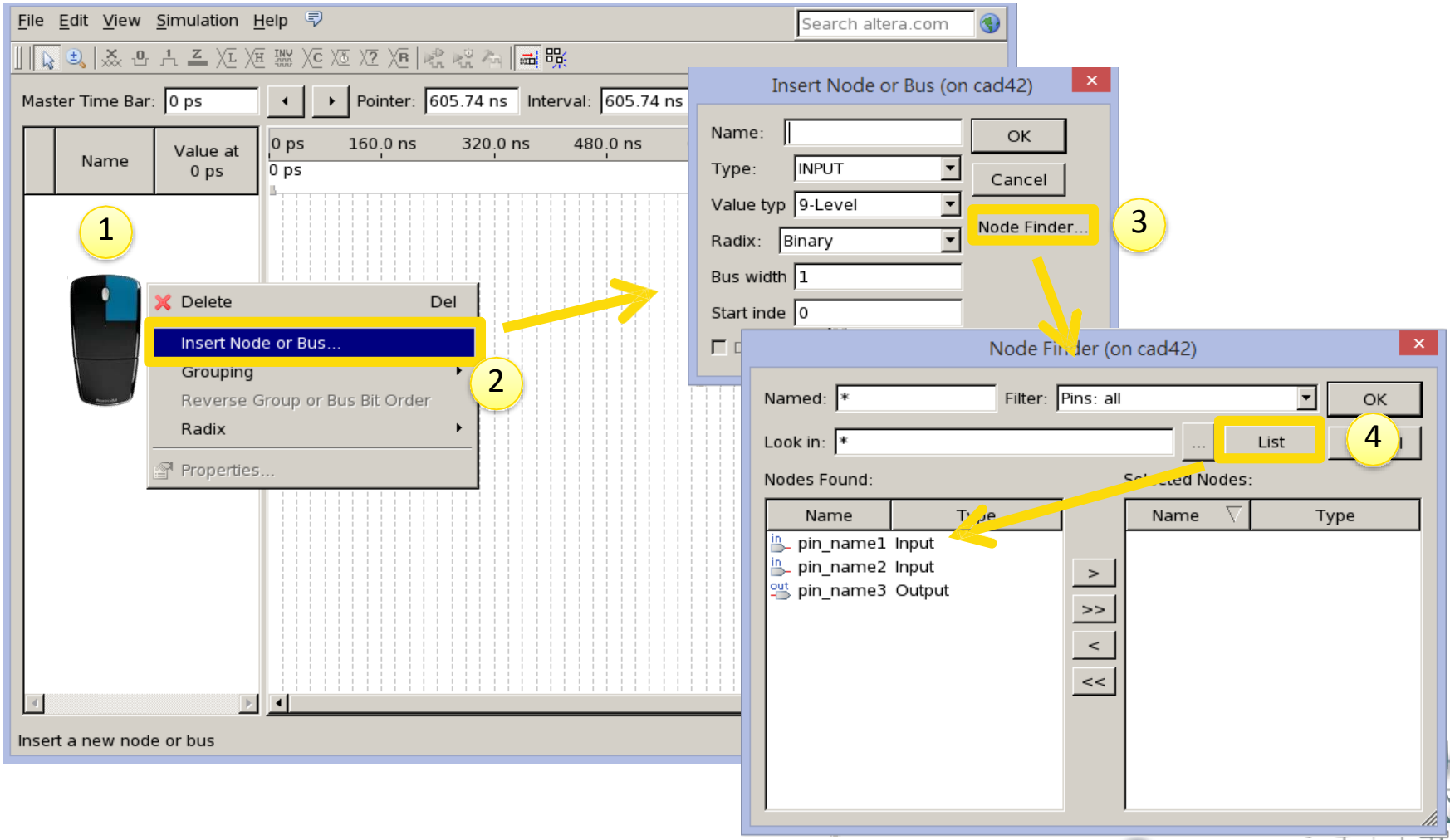
Compile



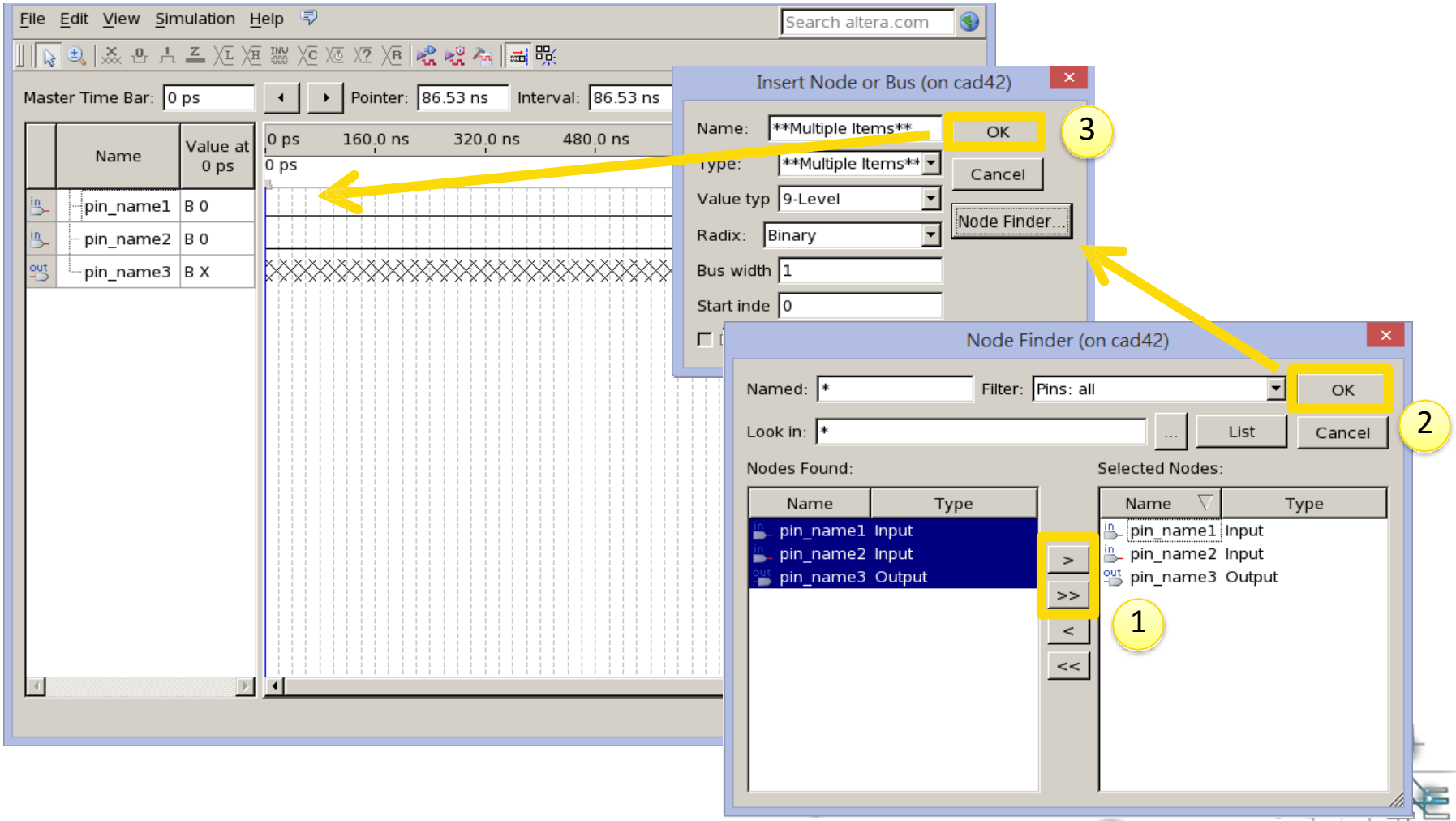
Set input signal



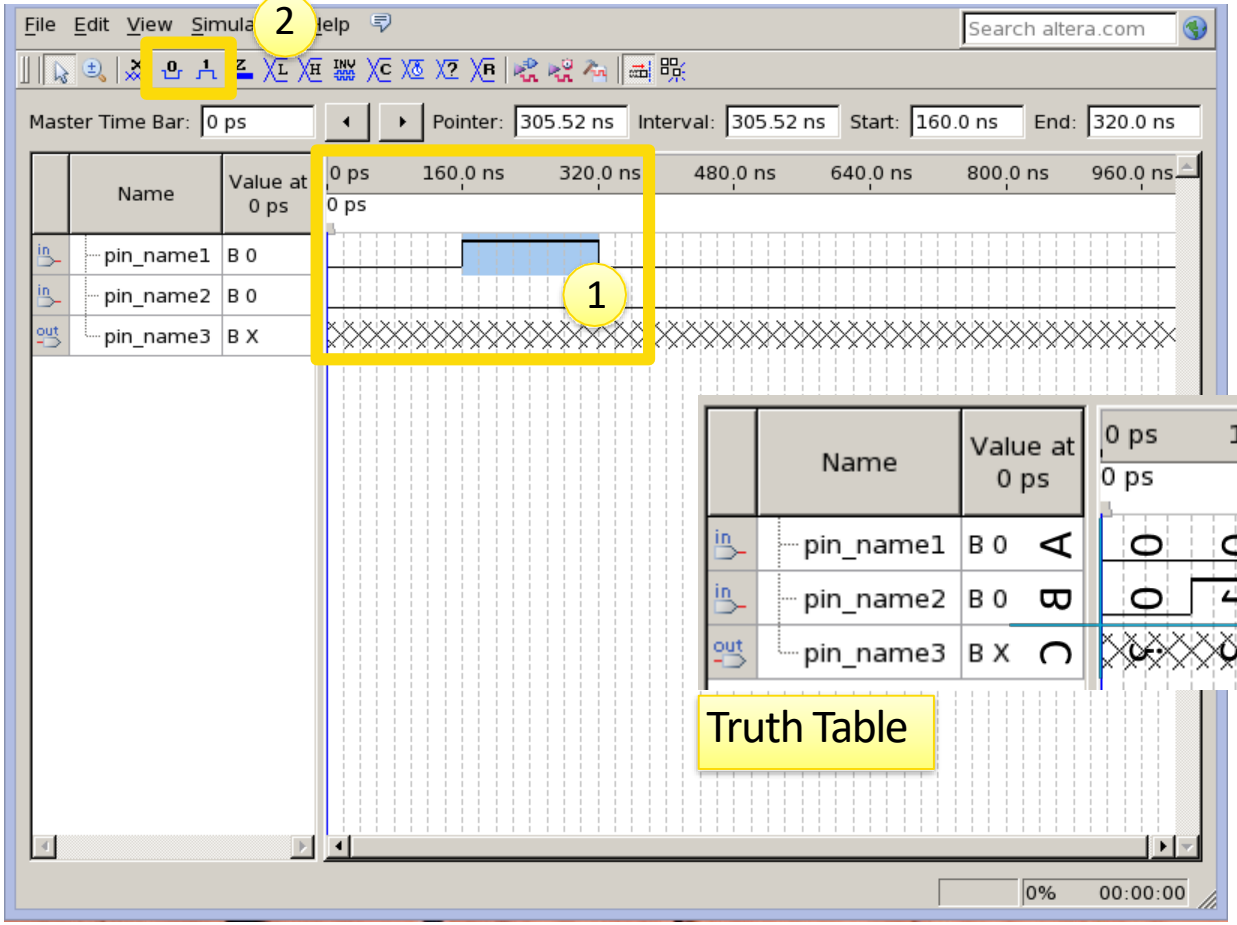
Set input signal



Set input signal



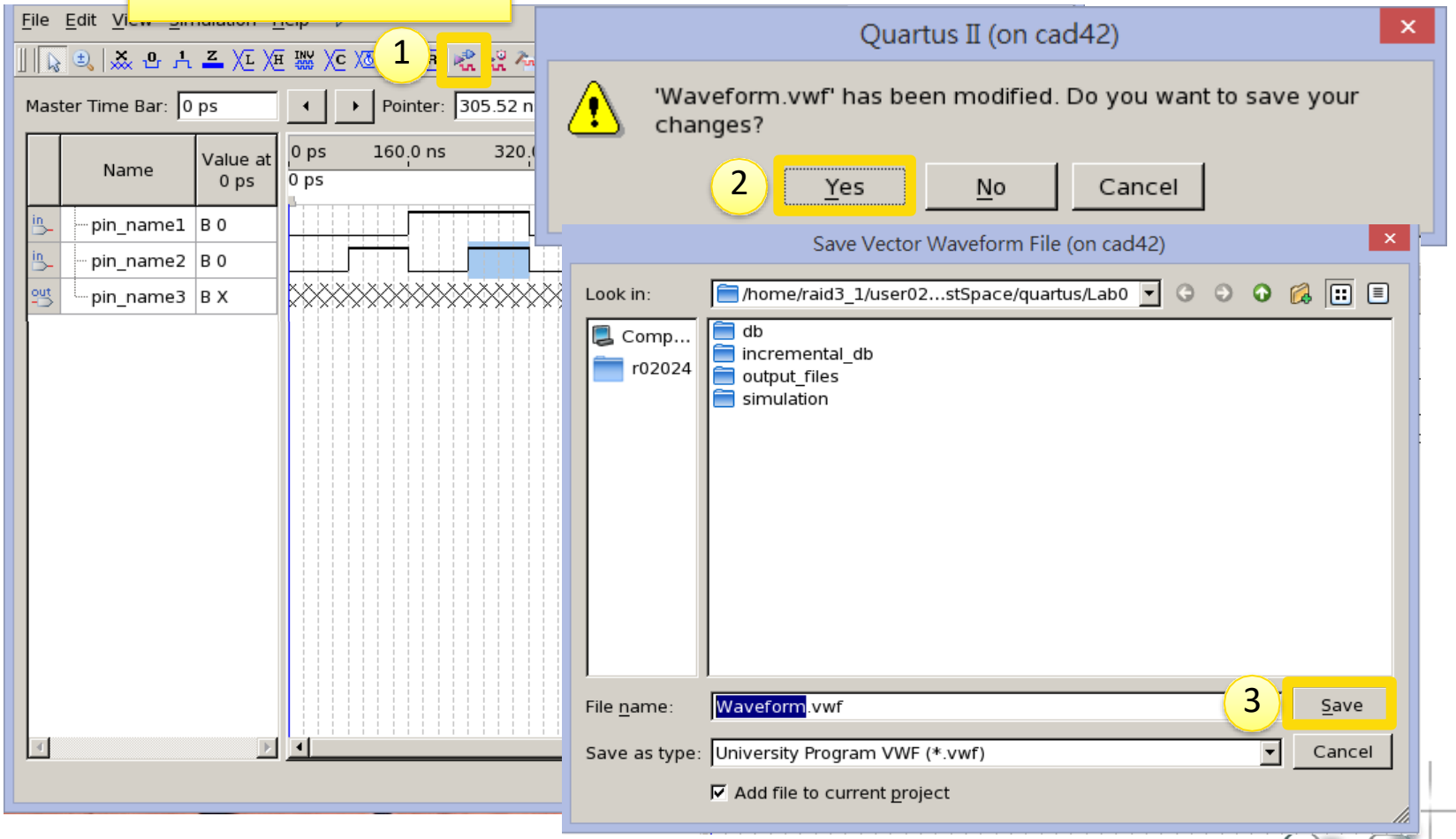
Set input signal



A	B	C
0	0	?
0	1	?
1	0	?
1	1	?

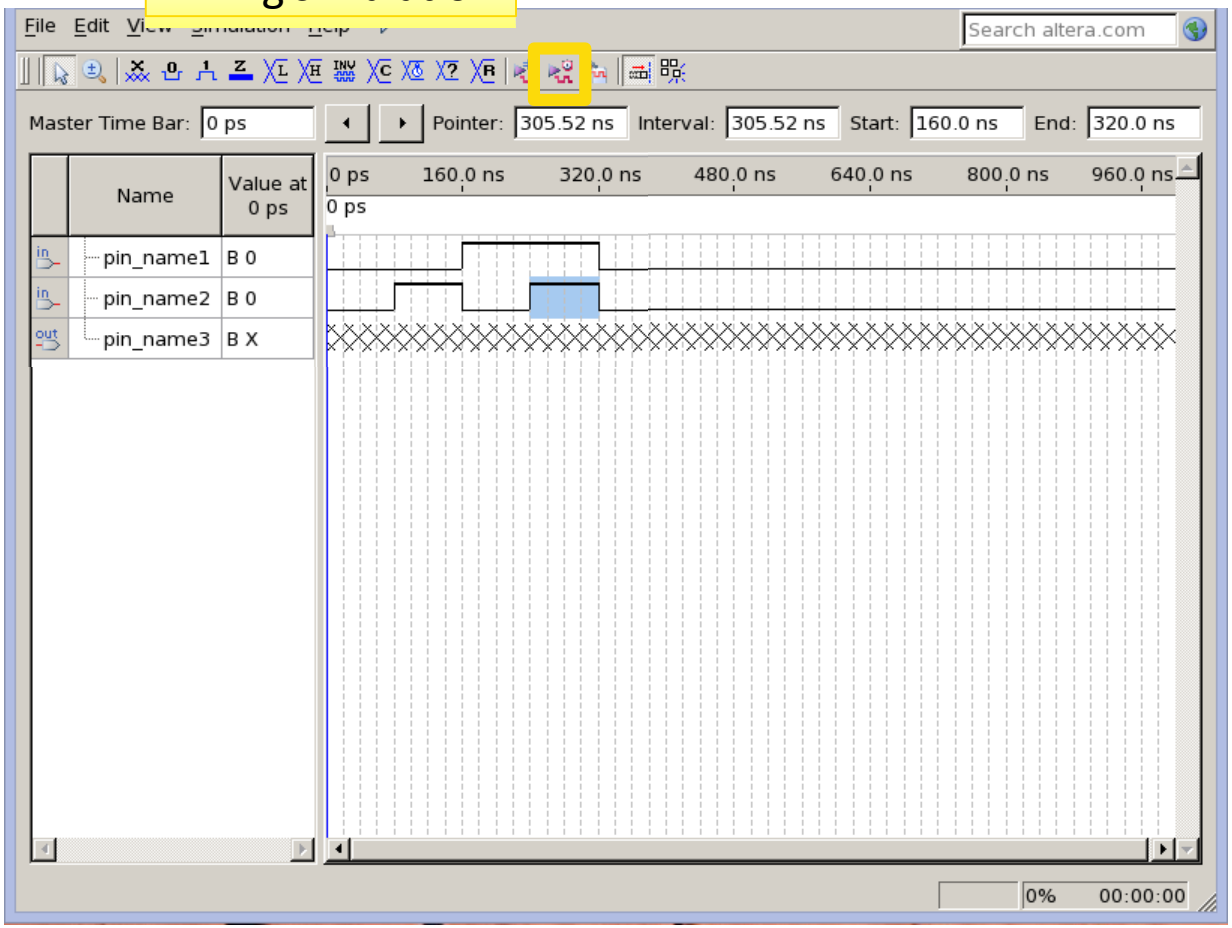
Simulation

functional simulation



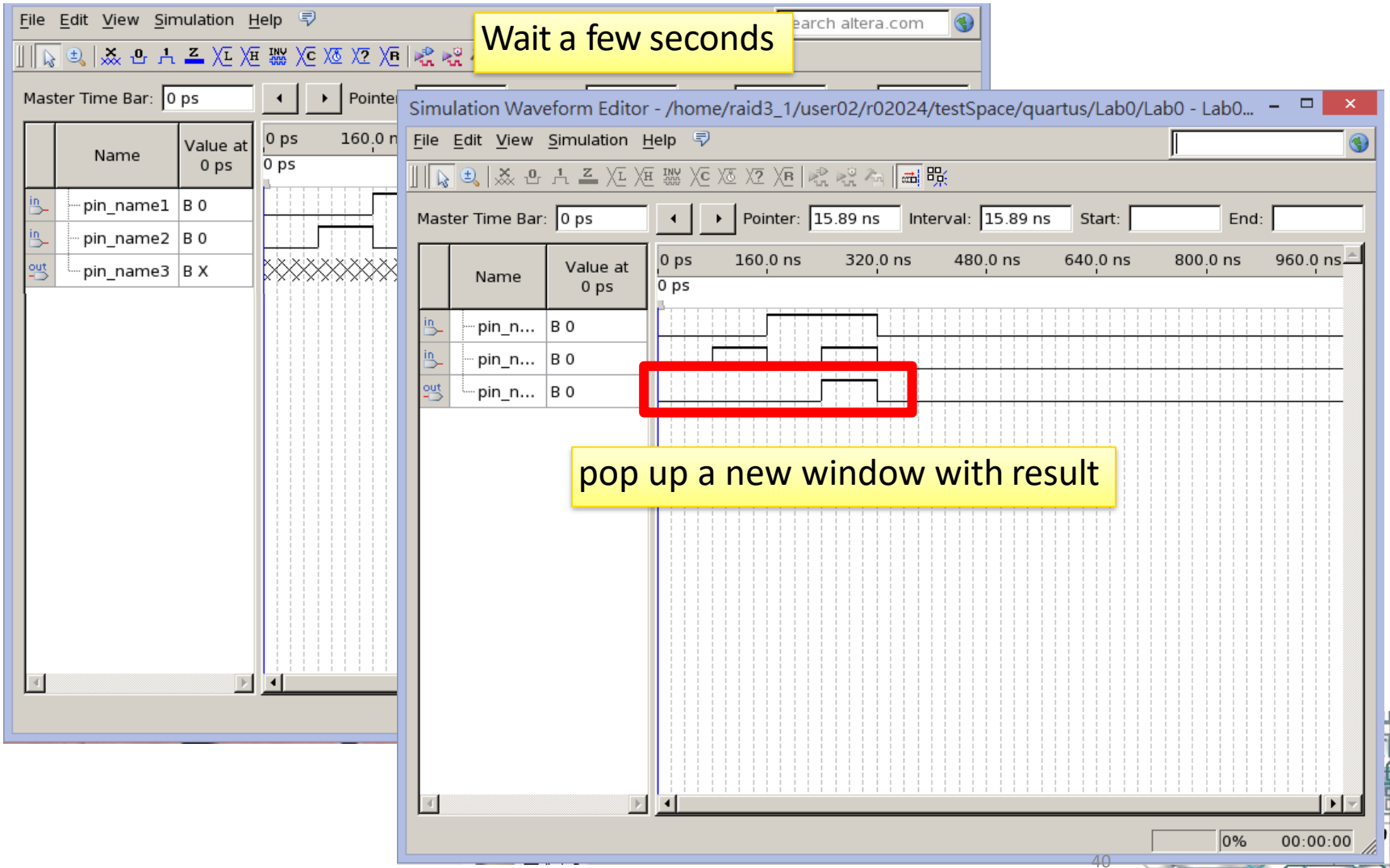
Simulation

Timing simulation

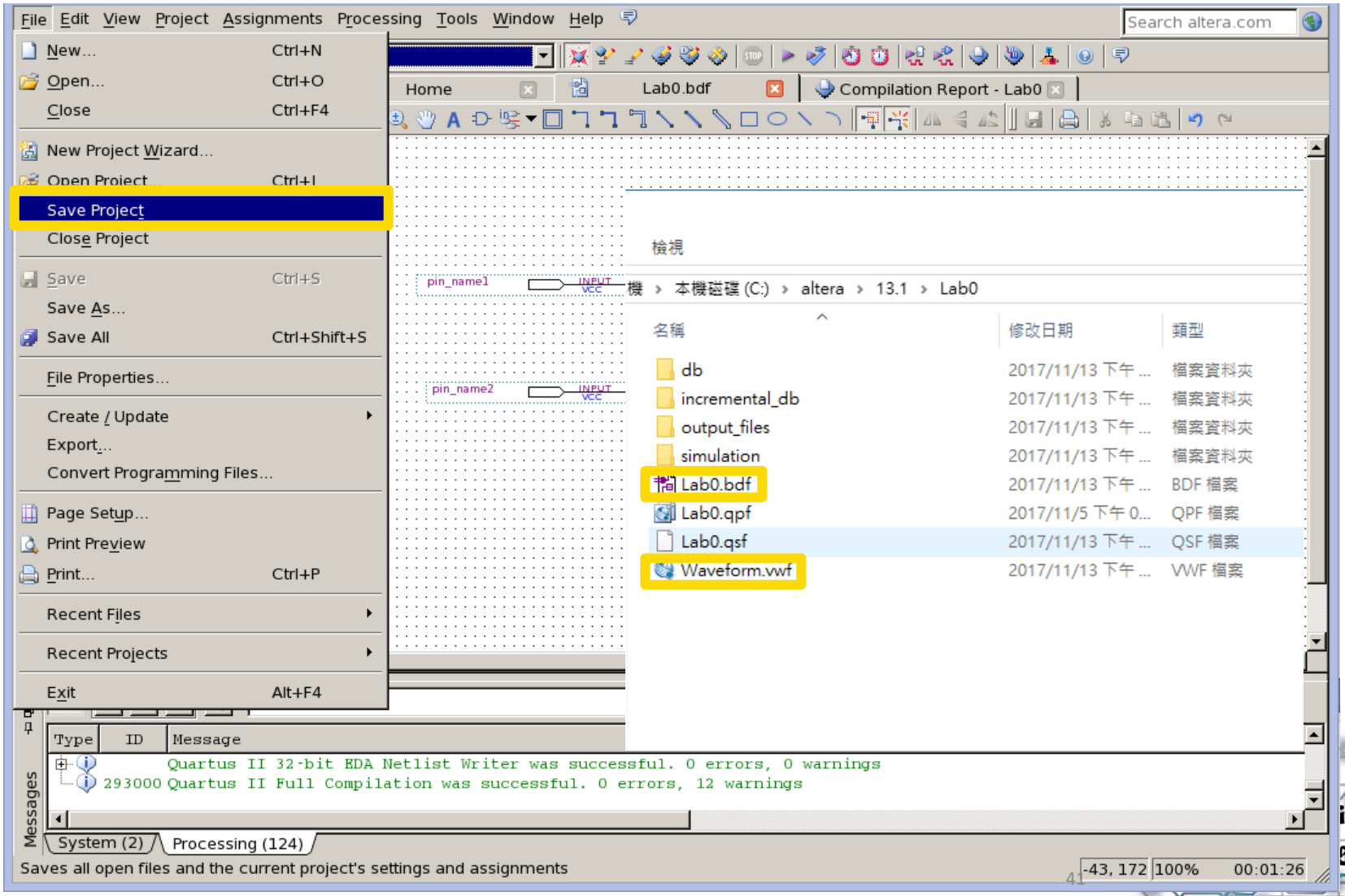


- You can run timing simulation to see gate delay and transition

Simulation


Wait a few seconds
File Edit View Simulation Help
Master Time Bar: 0 ps
Name
Value at 0 ps
pin_name1 B 0
pin_name2 B 0
pin_name3 B X
Simulation Waveform Editor - /home/raid3_1/user02/r02024/testSpace/quartus/Lab0/Lab0 - Lab0...
File Edit View Simulation Help
Master Time Bar: 0 ps
Pointer: 15.89 ns
Interval: 15.89 ns
Start:
End:
Name
Value at 0 ps
pin_n... B 0
pin_n... B 0
pin_n... B 0
0 ps
160.0 ns
320.0 ns
480.0 ns
640.0 ns
800.0 ns
960.0 ns
pop up a new window with result
0%
00:00:00

Save project



Lab1: 4 bits full adder [file: FullAdder4 (directory)]

Design Flow

Recall: Comb. Ckt Design flow

- Gate-level schematic design flow:
  1. Divide system - Divide big design to a system of small modules
     - For each small module, do the following steps:
  2. Truth table - Think thoroughly about your topic
  3. K-map - Simplify the logic (multi-output simplification)
  4. Boolean expression - The exact form you're going to make
  5. Draw the Ckt - Be sure meet the requirements (ex. gate count)
  6. Design with tools - Implement it (.bdf or .v)
     - You can pack partial circuit as sub-module by .bsf + .bdf
  7. Verify & debug - Test some typical patterns, simulate it by wave form

Step 1: Divide Your System

- Divide big design to system of small modules
  - We decide to build a 4-bit ripple adder with 4 identical full adders
    - => easier todesign

FIGURE 4-2
Parallel Adder for 4-Bit Binary Numbers

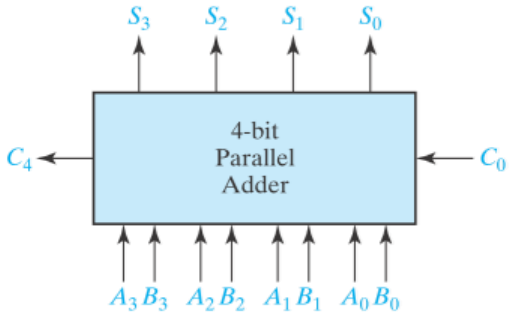
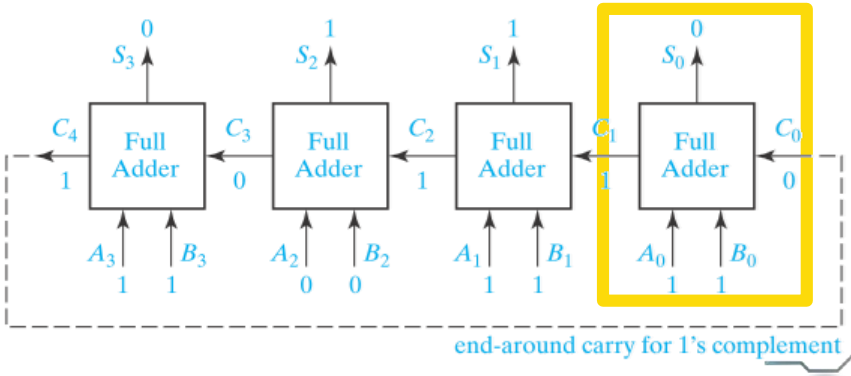


FIGURE 4-3
Parallel Adder Composed of Four Full Adders



Step 2: Truth Table

- For every small modules, do the steps: 2, 3, 4, 5, 6
  - But we only have to make a full adder in the tutorial
- Think thoroughly about your topic, list the logic

FIGURE 4-4
Truth Table for a Full Adder

X	Y	C_{in}	C_{out}	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Step 3: K-map Simplify

- Simplify the logic
  - Do multi-output simplification if needed

S

xy \ Ci	0	1
00	0	1
01	1	0
11	0	1
10	1	0

x⊕y

xy \ Ci	0	1
00	0	0
01	1	1
11	0	0
10	1	1

(x⊕y)&Ci

xy \ Ci	0	1
00	0	0
01	0	1
11	0	0
10	0	1

Co

xy \ Ci	0	1
00	0	0
01	0	1
11	1	1
10	0	1

or

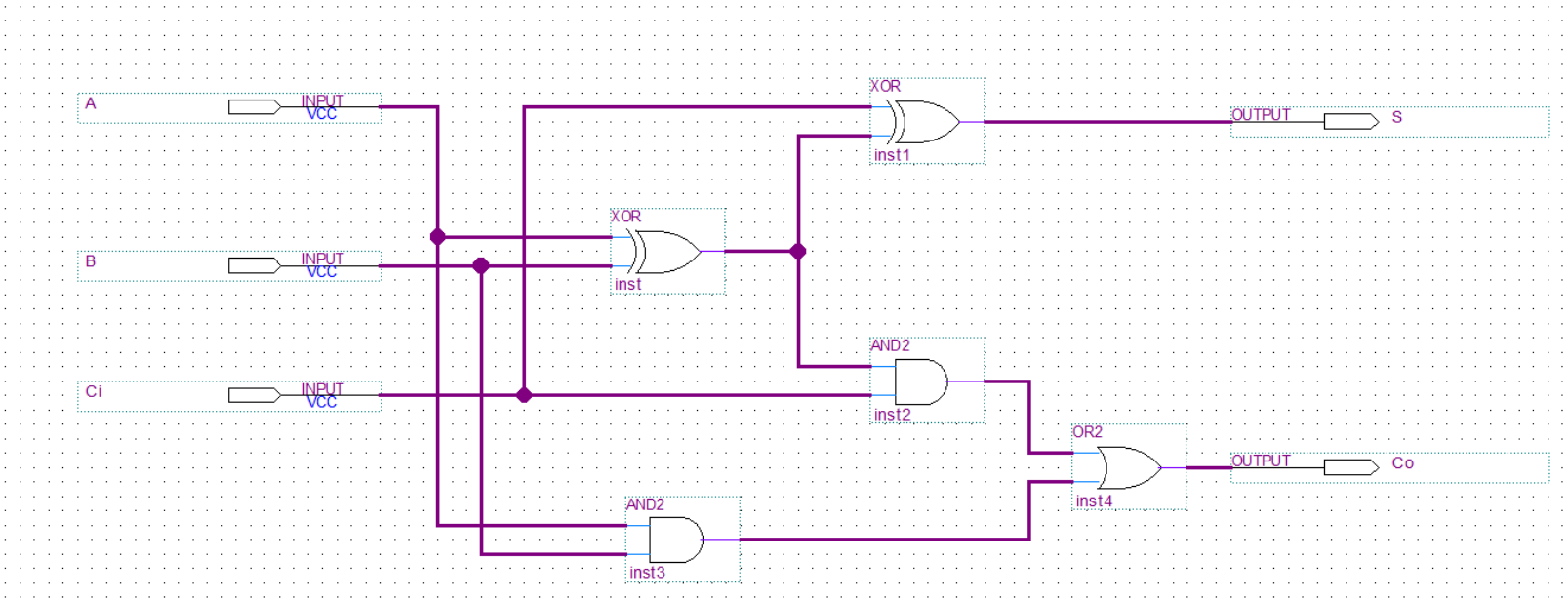
Co

xy \ Ci	0	1
00	0	0
01	0	1
11	1	1
10	0	1

$S = x \oplus y \oplus Ci$

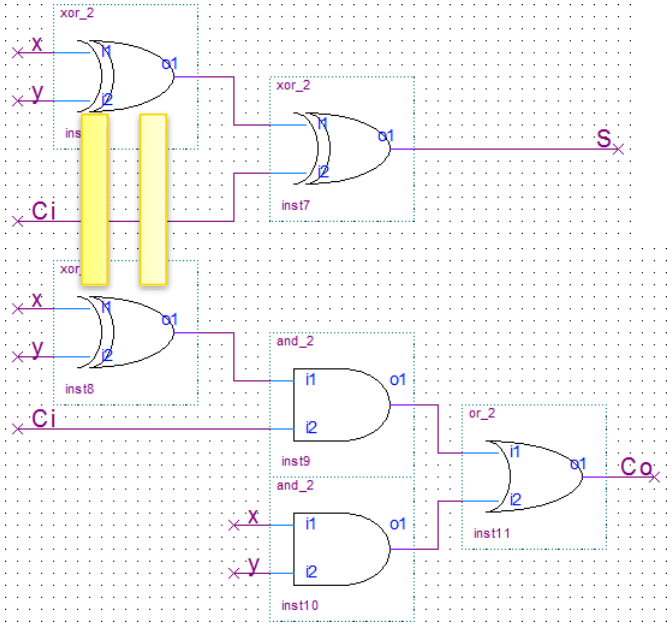
Step 4: Boolean Expression

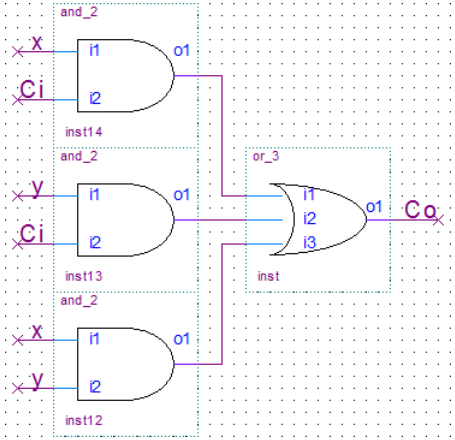
- The exact form you're going to make
  - $S = (x \oplus y) \oplus Ci$
  - $Co = (x \oplus y) \cdot Ci + x \cdot y$
    - or
  - $Co = x \cdot Ci + y \cdot Ci + x \cdot y$



Step 5: Pre-Draw & Check

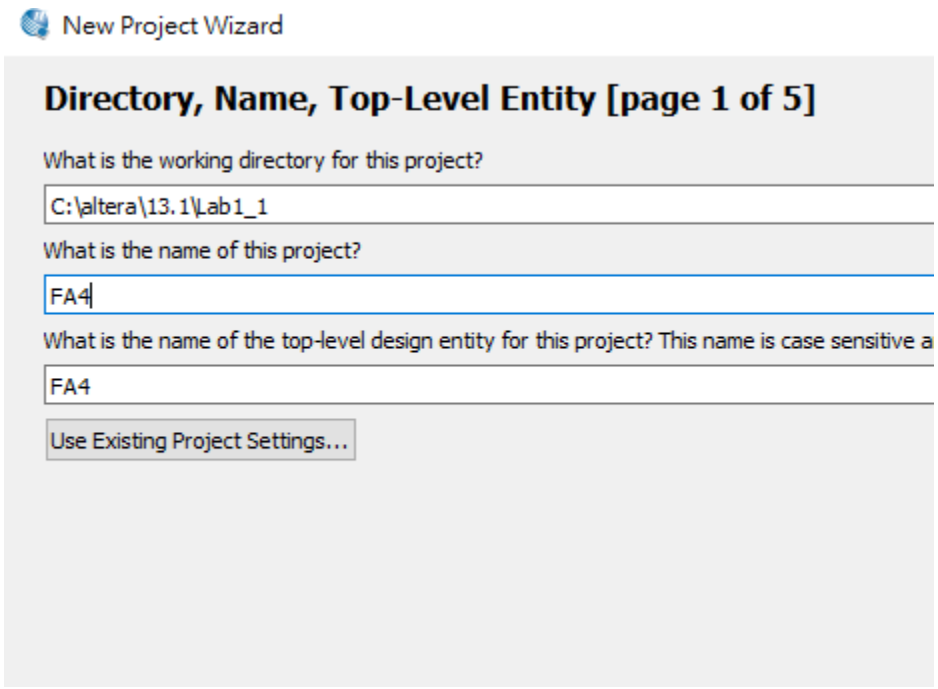
- Be sure meet the requirements
  - ex. gate count, how many levels
  - S = (x⊕y)⊕Ci
  - Co = (x⊕y)·Ci + x·y
    - or
  - Co = x·Ci + y·Ci + x·y





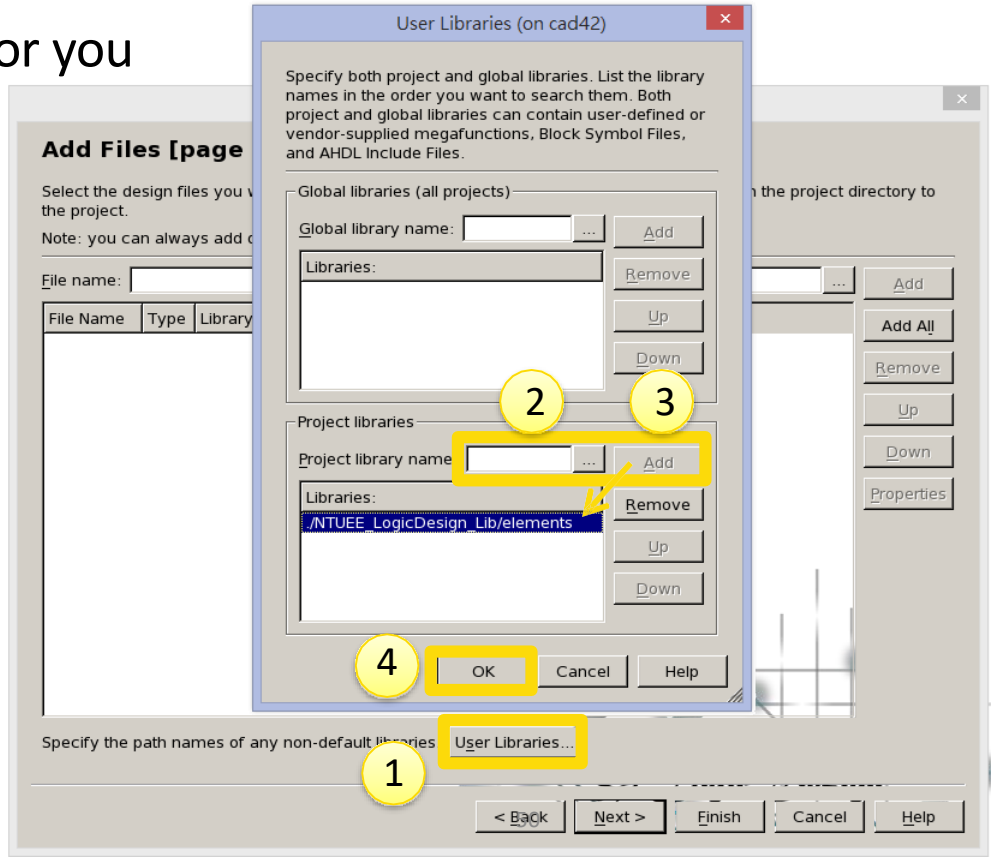
Step 6: Quartus Design

- Launch Quartus II
- Create a new project named FA4
  - File -> new project wizard



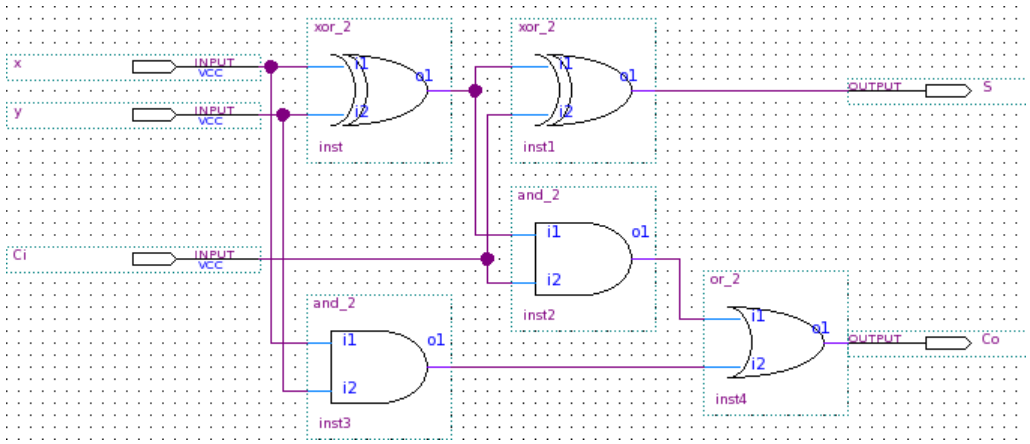
Important things about this project

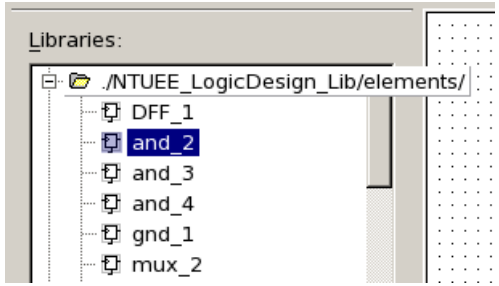
- This project is built with top design named “FA4”
- In all the Labs, we restrict you to use logic gates TAprovided only
  - TA already put them as library for you
  - If you want to set it yourself in the future, do this → when building the project



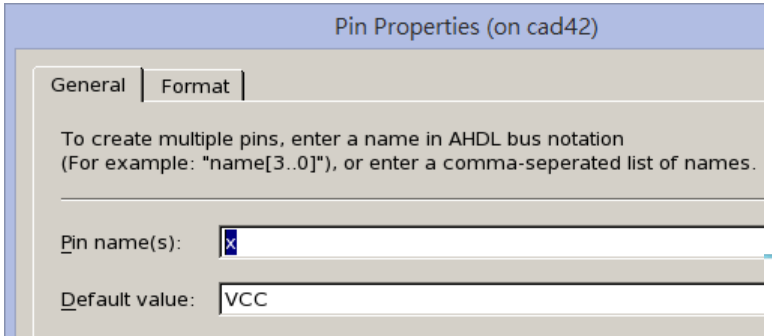
Build the FullAdder1.bdf

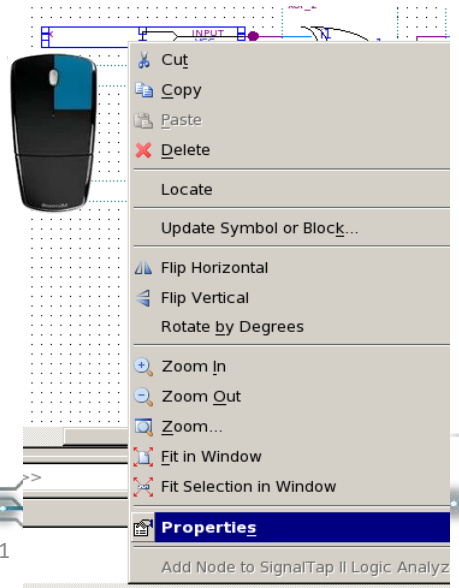
- File→New…→Block Diagram/Schematic File→OK
  - Implement it only use gates under NTUEE_LogicDesign_Lib





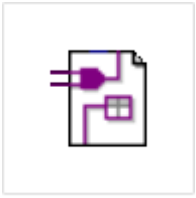
- Important hint!!
  - right click on the I/O pin→Properties to name it
  - The name of pin is important!!





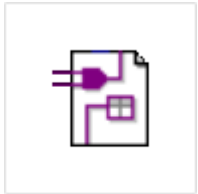
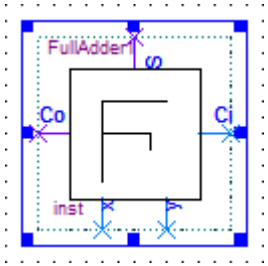
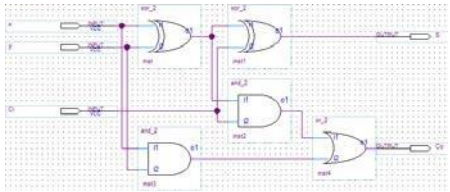
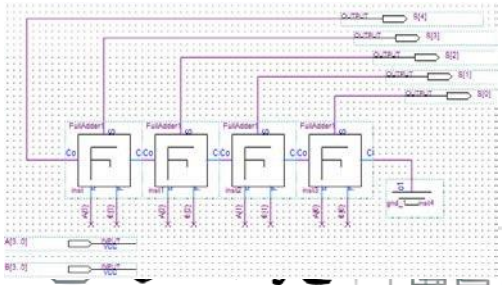
Make FullAdder1 a “Module”

- We want to use the “FullAdder1” 4 times, we can declare it as a module (like C++ class) and generate many instance!!
- All you need is a .bsf file

	Declaration	Implementation	Instance
Quartus Schematic Module	Module Block Symbol File with I/O port info.  FullAdder1.bsf	Ckt Design Block diagram File  FullAdder1.bdf	Grab an instance block to another bdf file ex. FA4 to use it
C++ Object	Object Class (header file) with data, interface  EEmitation.h	Object Class (source file)  EEmitation.cpp	Use class included by header in another cpp ex. main.cpp

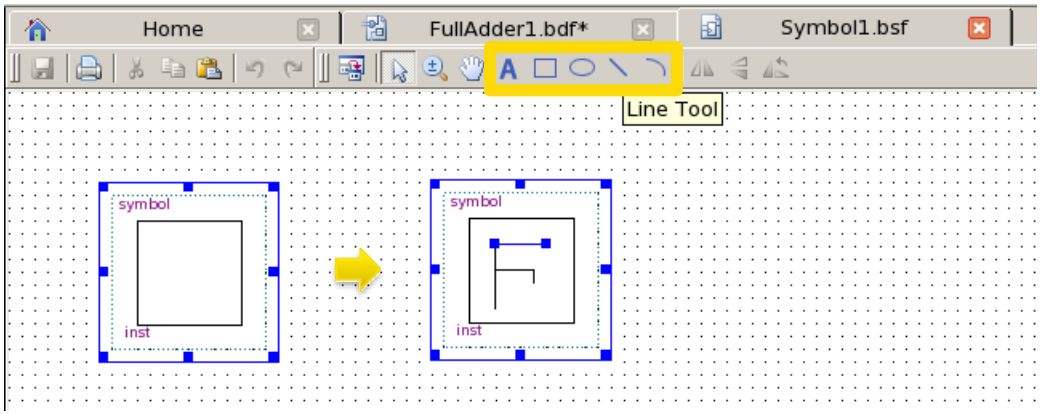
Make FullAdder1 a “Module”

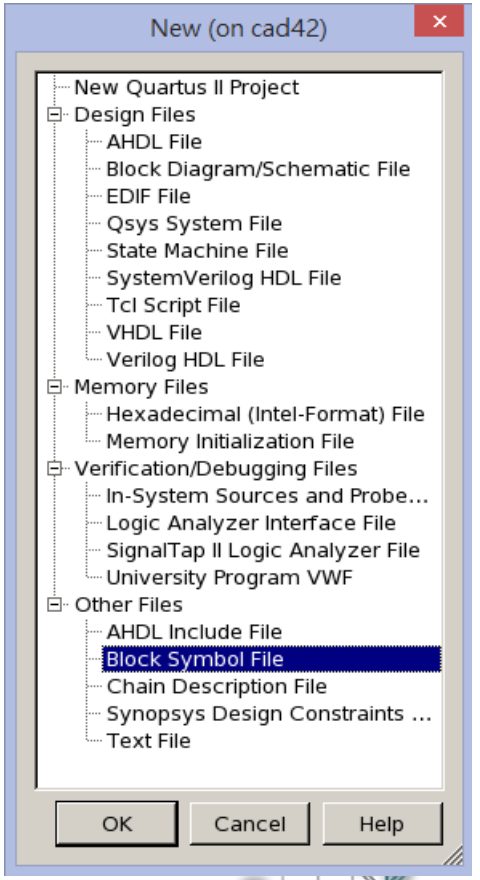
- We want to use the “FullAdder1” 4 times, we can declare it as a module (like C++ class) and generate many instance!!
- All you need is a .bsf file

	Declaration	Implementation	Instance
Quartus Schematic Module	Module Block Symbol File with I/O port info.  FullAdder1.bsf	Ckt Design Block diagram File  FullAdder1.bdf	Grab an instance block to another bdf file ex. FA4 to use it
			

Draw a FullAdder1.bsf (1/3)

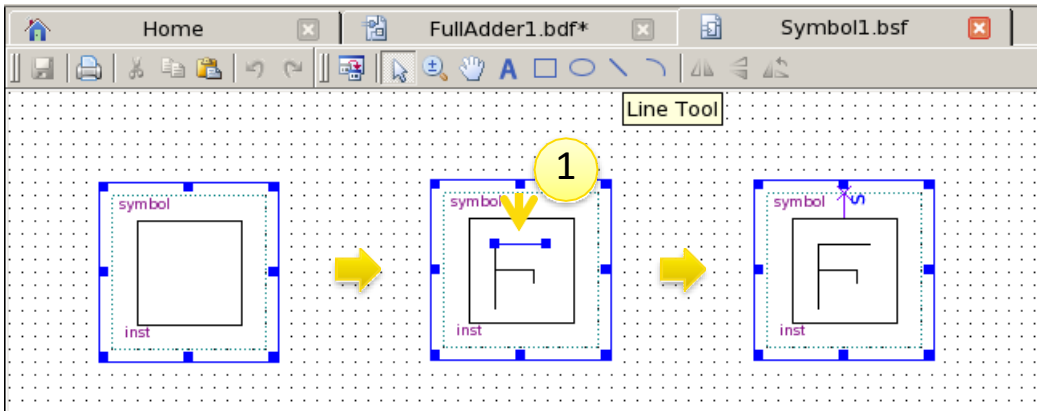
- File→ New…→Block Symbol File→OK
- Draw it as you wish by drawing tools

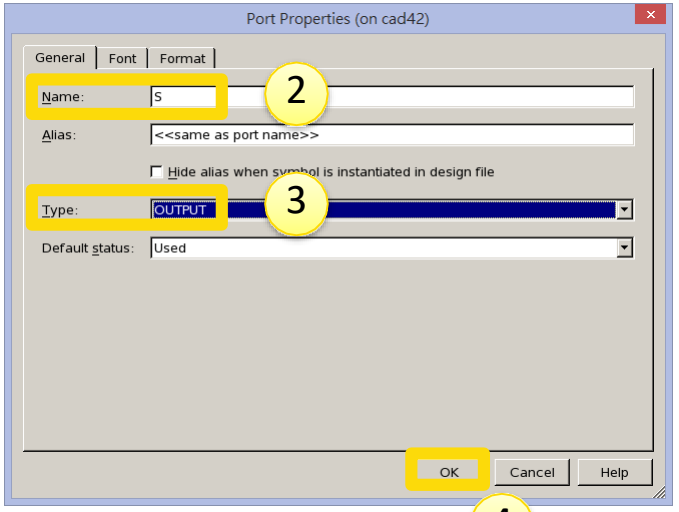




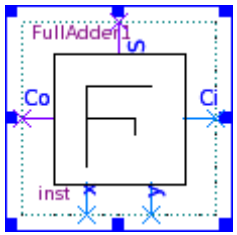
Draw a FullAdder1.bsf (2/3)

- Drag at the edge to add pins
- Name the pin properly and choose I/O type



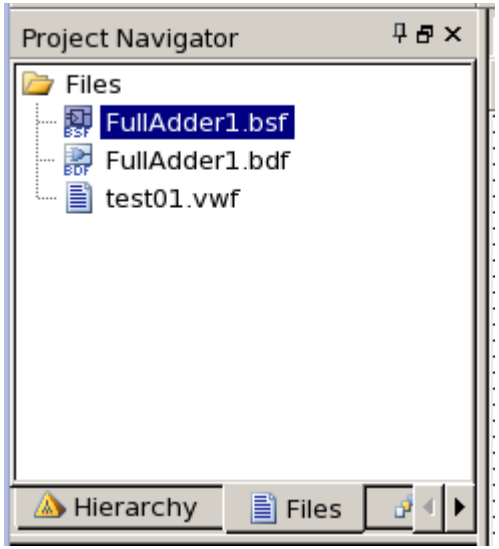


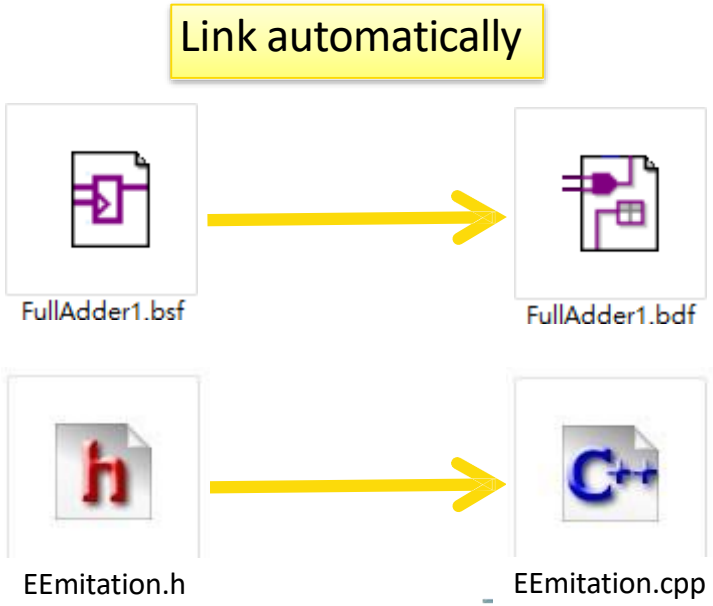
- Repeat to add all pins properly



Draw a FullAdder1.bsf (3/3)

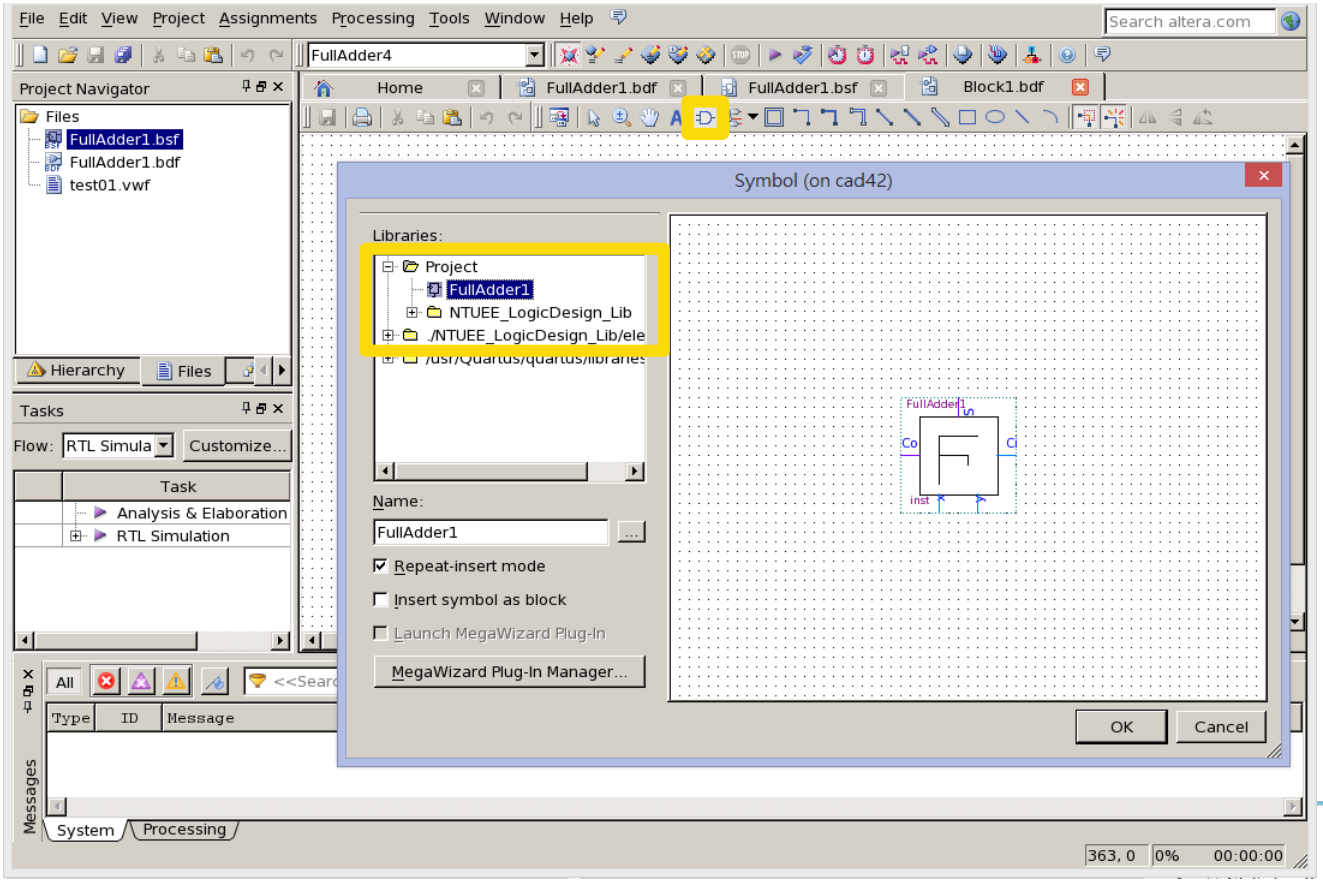
- Save this .bsf as FullAdder1.bsf
  - Important!! Names are mattered for Quartus
  - The .bsf will search for implementation file with the same name





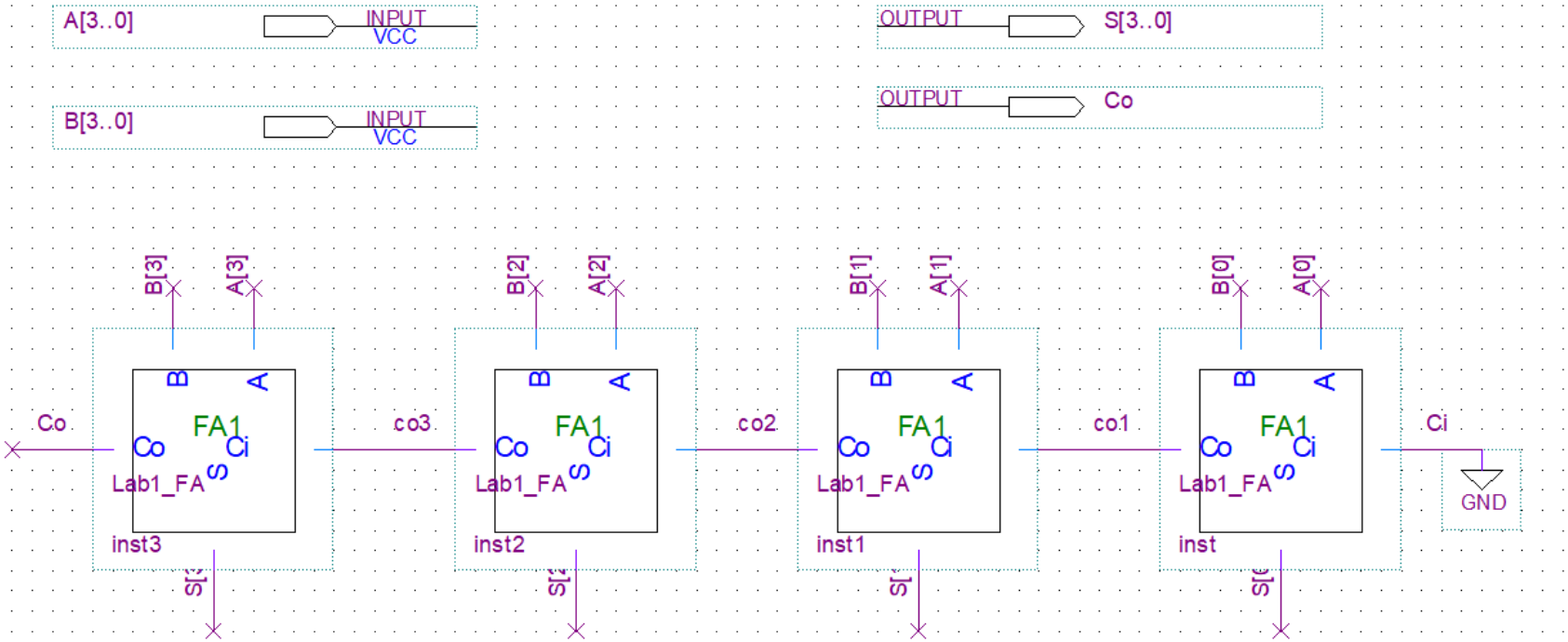
The Top Design: FA4.bdf

- File→New…→Block Diagram/Schematic File→OK
- From now on, you can use the “FullAdder1” module



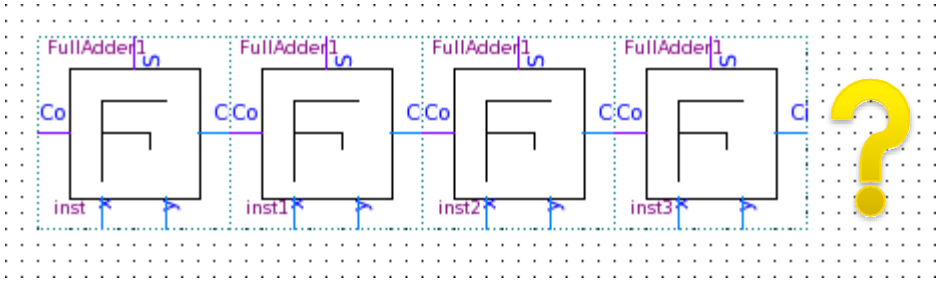
The Top Design: FA4.bdf

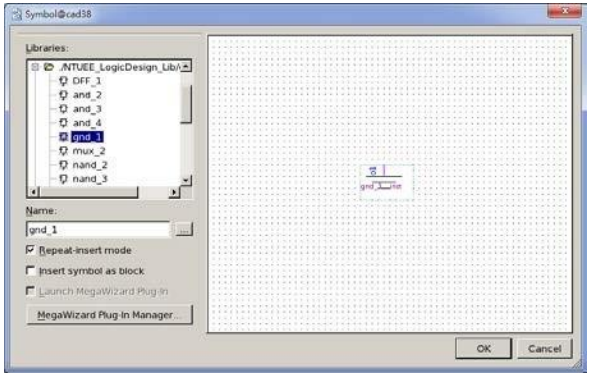
- File→New…→Block Diagram/Schematic File→OK
- From now on, you can use the “FullAdder1” module

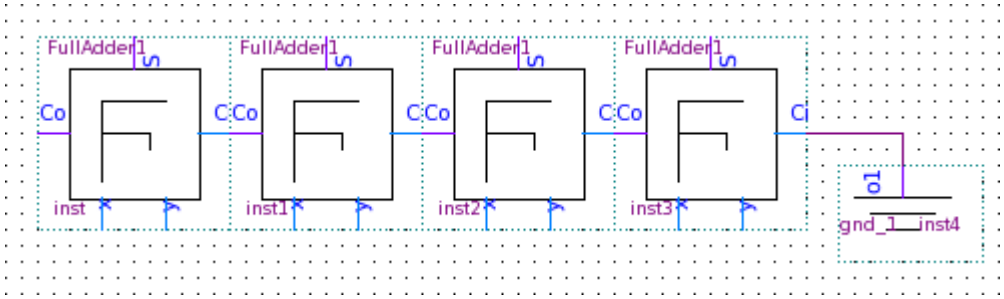


0 and 1, Ground and VCC

- You can
  - use ground (gnd_1) as 0
  - use voltage circuit (vcc_1) as 1

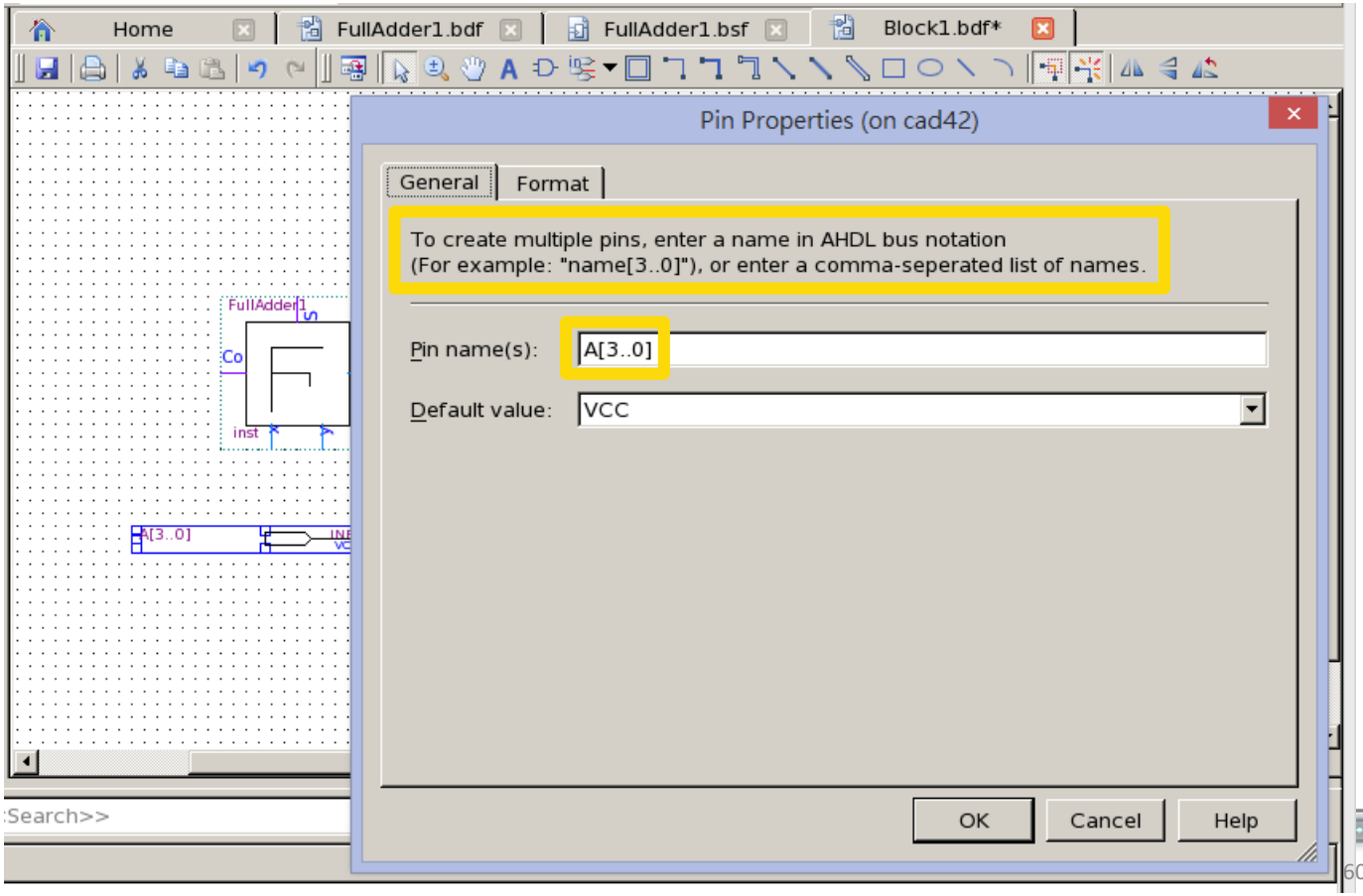






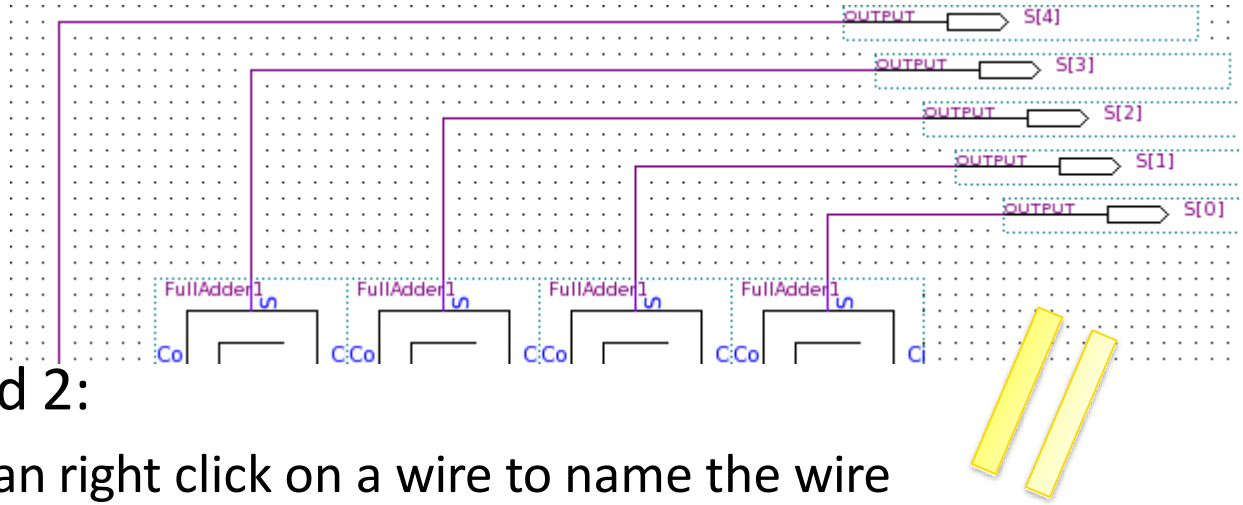
Array of pins / wires

- By naming the I/O pins as xxx[(N-1)..0], it would generate N pins with the name: xxx[0], xxx[1], …, xxx[N-1]

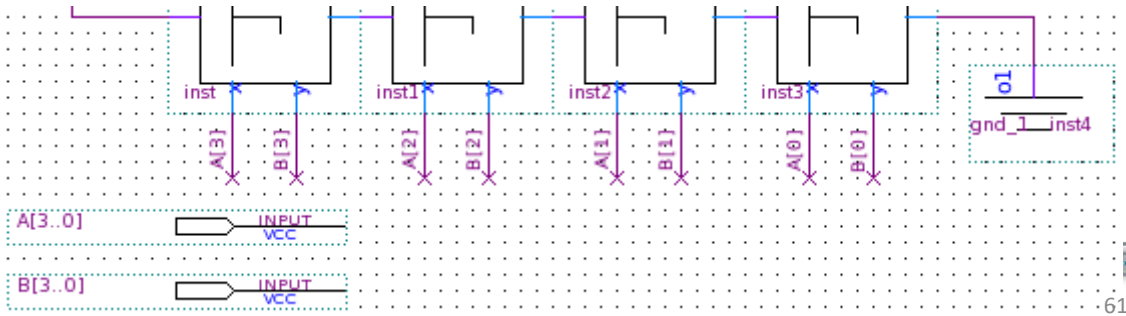


Array of pins / wires

- Method 1:
  - You can expand the pins, and link each pin with proper wire



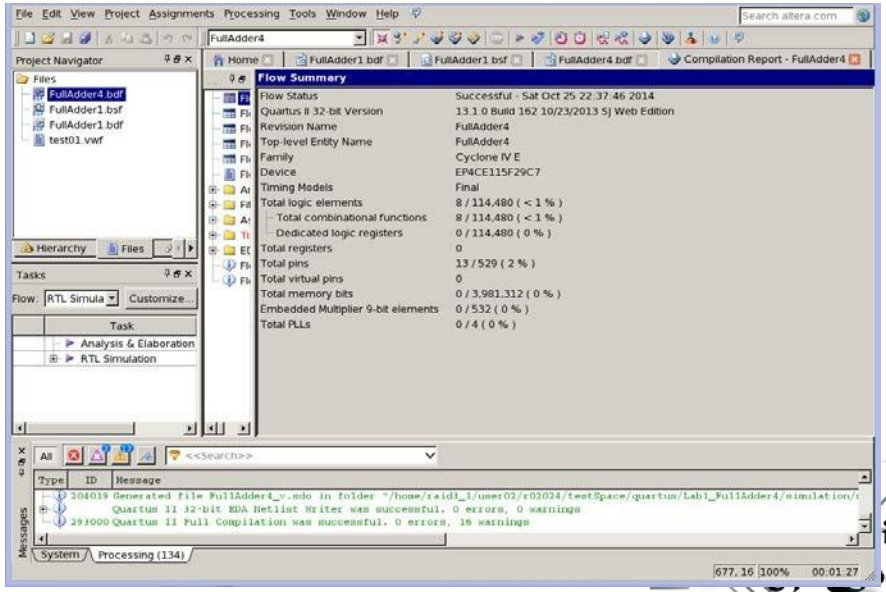
- Method 2:
  - You can right click on a wire to name the wire
  - Wires / pins with the same name "are linked implicitly"



The Top Design: FA4.bdf

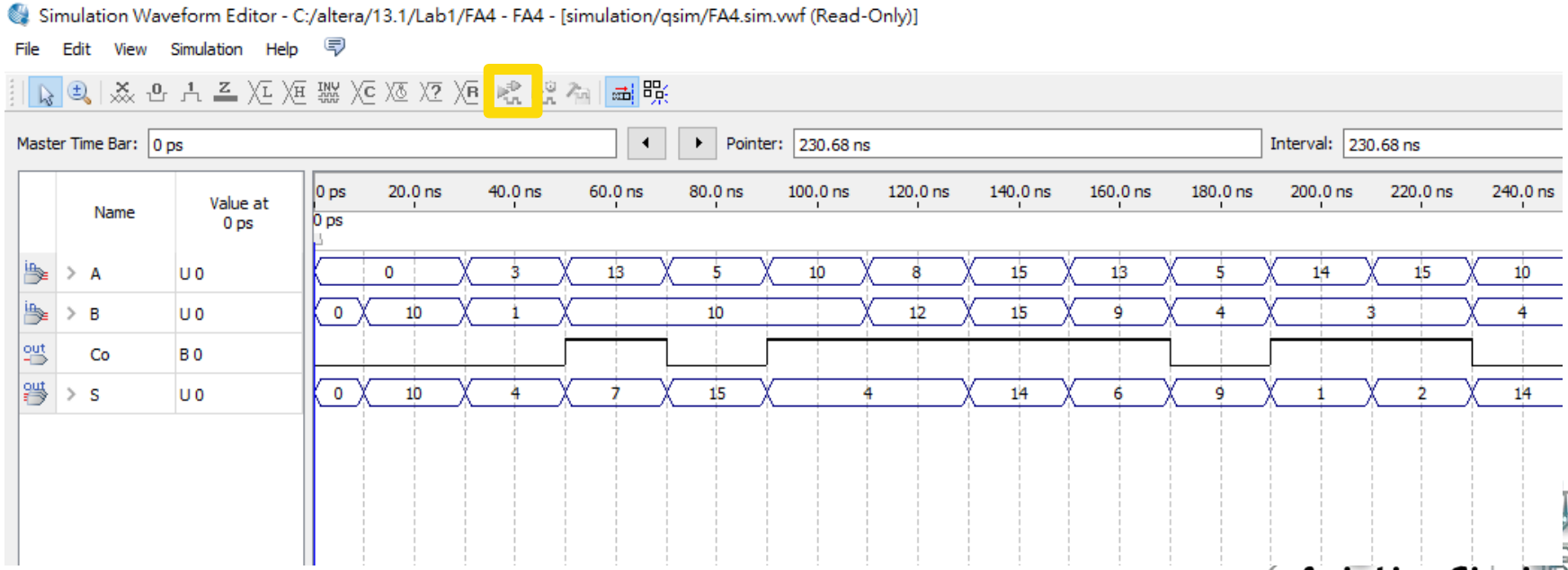
- Remember to save this bdf as FA4.bdf
  - The “name” that our top design is set when project created
- File→Save poject

- Start Compilation
  - Check no error exists
- File→Save poject



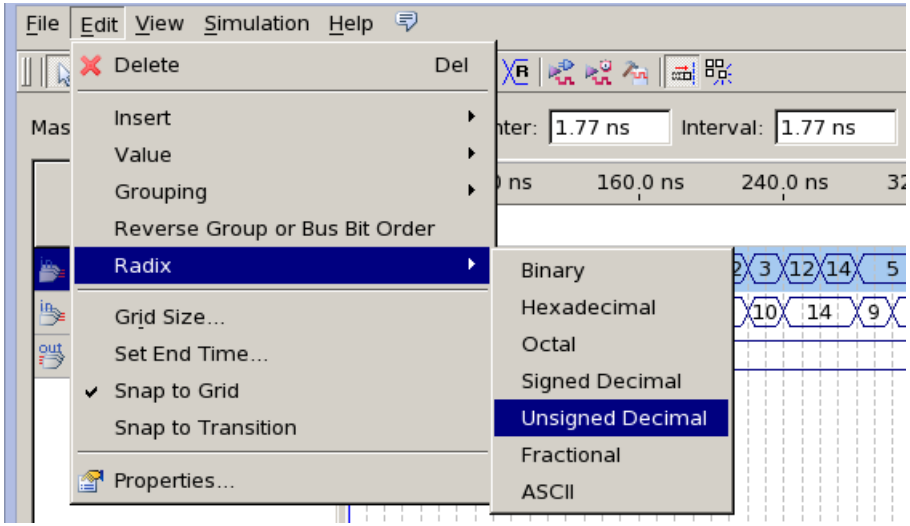
Step 7: Verify & Debug

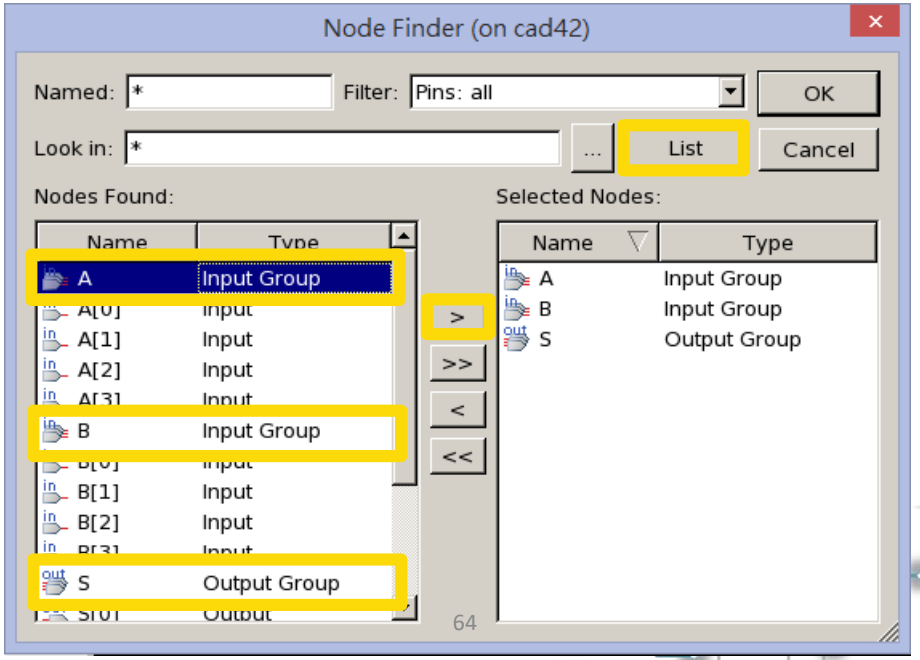
- Test some typical patterns, simulate it by .vwf
- We are “requested” (given) a test01.vwf file
  - We must follow the I/O pins name of it
  - And pass all the patterns



vwf Advanced

- How can we make a vwf like this?
  - Follow steps in p.32 – p.40
  - You can pick “wire group” when using node finder
    - Right click →Insert Node or Bus…
      →Node Finder… →List
  - You can select the interval and ctrl+alt+r to generate random value
  - You can set radix to change base





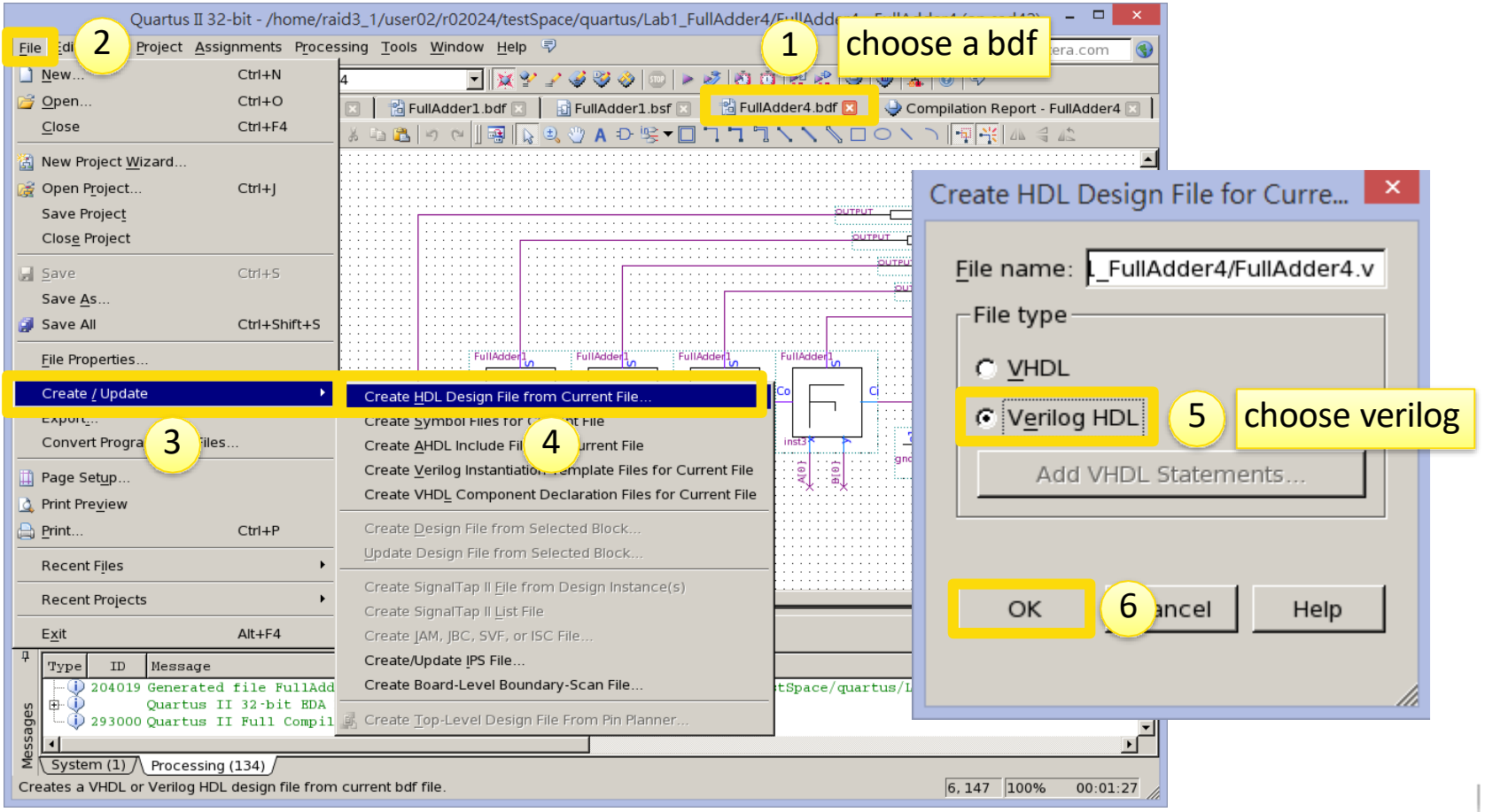
Quick Preview

Verilog HDL - Comb. Gate Level design

What is Verilog HDL(Hardware Description Language)?

- Key features of Verilog
  - **Multiple levels of abstraction**
    - Behavioral
    - Functional (RTL:Register Transfer Level)
    - Structural (Gate-Level)
  - Model the timing of the system
  - Express the concurrency
  - Verify the design
- Why not C++ ?
  - Not natively used for ckt design
  - Natively an imperative programming language
  - Ckt is parallel anytime anywhere
    - Every time every signals change their values, we need to trace them & interact with each other
- But be careful that our computer is still imperative
  - => All the parallel performance is "simulated"
  - Verilog is a language easy to "design HW" & "simulate it"

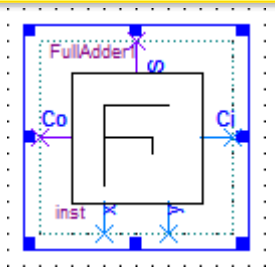
Generate Verilog Design File from Schematic



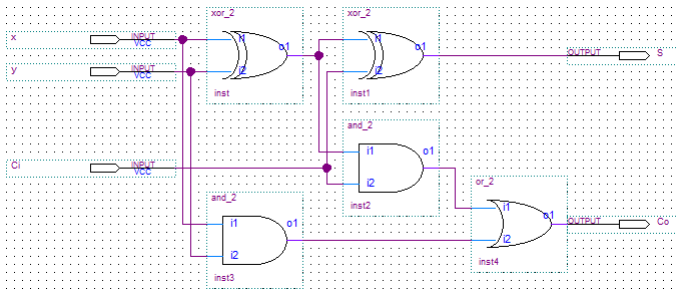
An Example - FullAdder

- Interconnections of logic elements

Module interface declaration

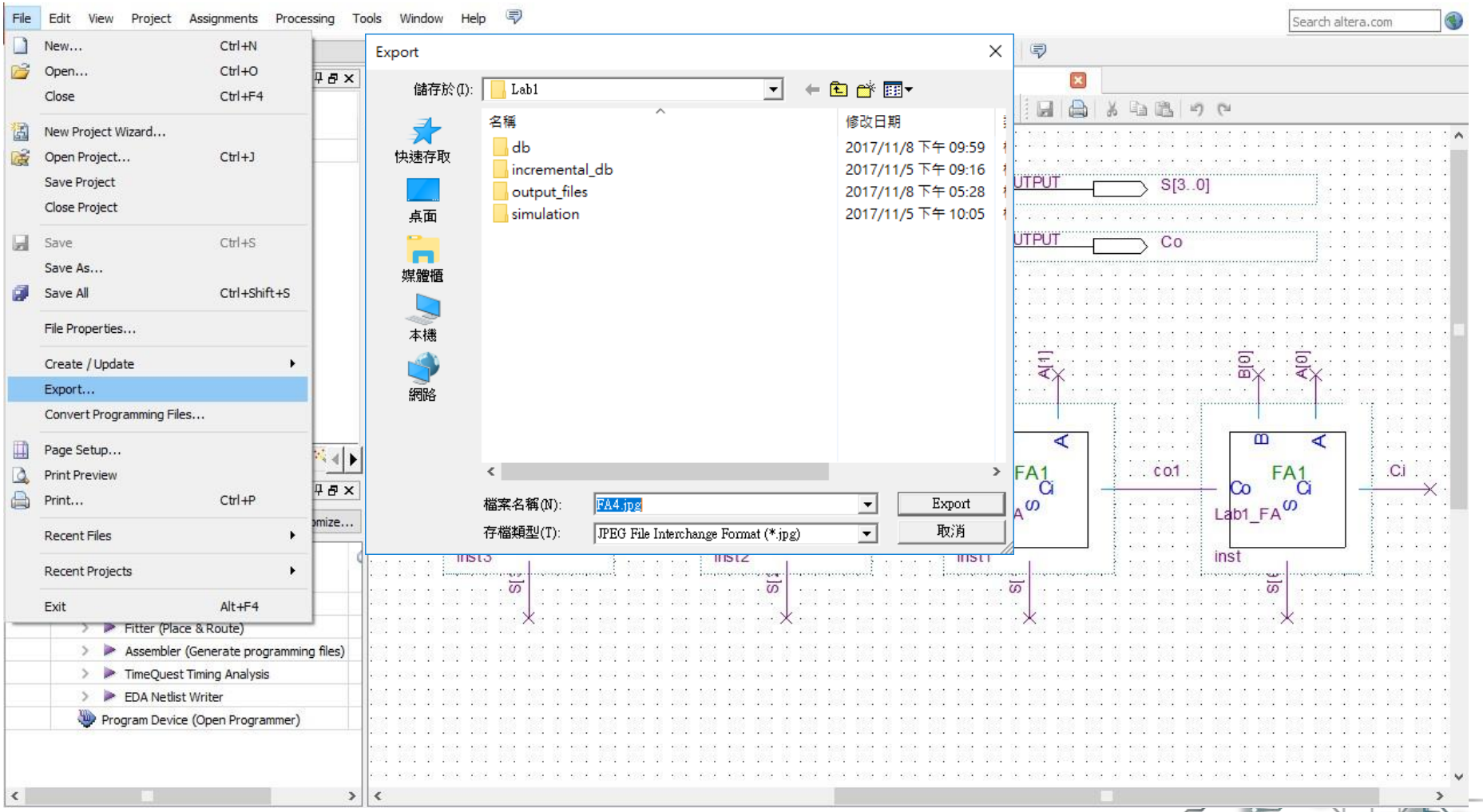



```
module FullAdder1(Ci,x,y,S,Co);
input wire  Ci;
input wire  x;
input wire  y;
output wire S;
output wire Co;

wire    SYNTHESIZED_WIRE_4;
wire    SYNTHESIZED_WIRE_2;
wire    SYNTHESIZED_WIRE_3;

xor_2   b2v_inst(
    .i1(x),
    .i2(y),
    .o1(SYNTHESIZED_WIRE_4));

xor_2   b2v_inst1(
    .i1(SYNTHESIZED_WIRE_4),
    .i2(Ci),
    .o1(S));

and_2   b2v_inst2(
    .i1(SYNTHESIZED_WIRE_4),
    .i2(Ci),
    .o1(SYNTHESIZED_WIRE_2));

and_2   b2v_inst3(
    .i1(x),
    .i2(y),
    .o1(SYNTHESIZED_WIRE_3));

or_2    b2v_inst4(
    .i1(SYNTHESIZED_WIRE_2),
    .i2(SYNTHESIZED_WIRE_3),
    .o1(Co));

endmodule
```



Descripting the interconnections

Generate block diagram image

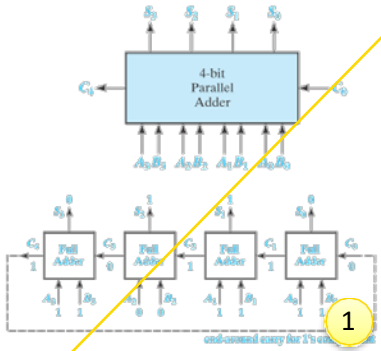


Summary

Schematic Combinational Circuit Design

General Design Flow

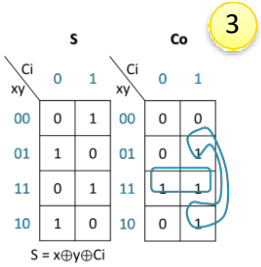
1. Divide system
2. Truth table
3. K-map
4. Boolean expression
5. Draw it first
6. Design with tools
7. Verify & debug



1

X	Y	C_{in}	C_{out}	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

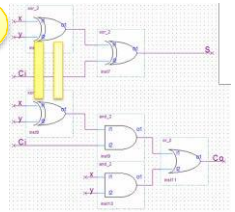
2



3

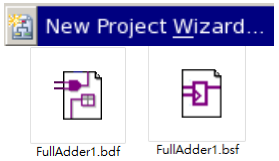
4
- S = (x⊕y)⊕Ci
- Co = (x⊕y)·Ci + x·y

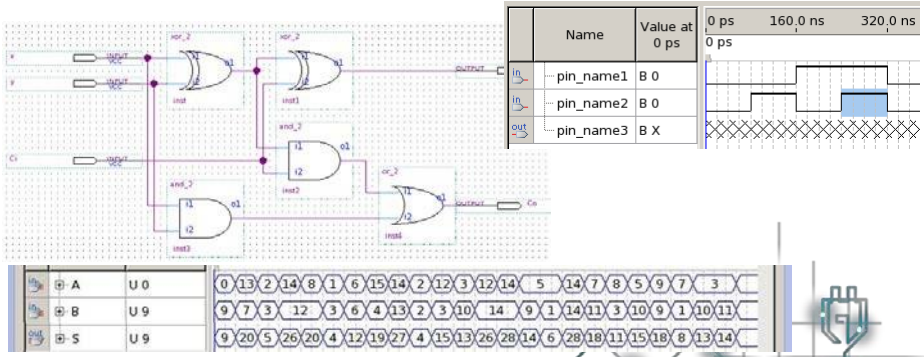
5



Quartus II Project Flow

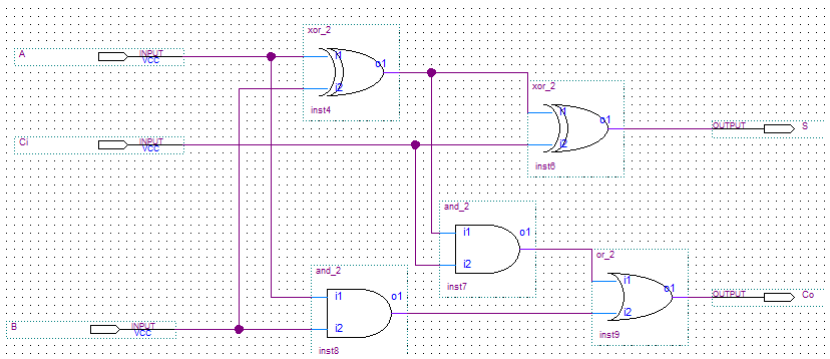
1. Launch Quartus II software
2. Build a Quartus project
3. Realize your circuit in .bdf
   - .bsf as modules declaration
4. Compile your design
5. Import / build a .vwf file as testbench
6. Run functional simulation





Schematic v.s. HDL

Schematic Design
(Block Diagram Design)



- **Visualizing**, easy to realize the hierarchical **architecture**
- Easy for beginners to catch up
- But hard to scale up for large designs
- Hard for programs to read

HDL Design
(Hardware description language)

```
module fullAdder1(
  input    Ci,
  input    A,
  input    B,
  output   S,
  output   Co
  );

  wire  WIRE_4;
  wire  WIRE_2;
  wire  WIRE_3;

xor_2 b2v_inst4(
    .i1(A),
    .i2(B),
    .o1(WIRE_4));

xor_2 b2v_inst6(
    .i1(WIRE_4),
    .i2(Ci),
    .o1(S));

and_2 b2v_inst7(
    .i1(WIRE_4),
    .i2(Ci),
    .o1(WIRE_2));

and_2 b2v_inst8(
    .i1(A),
    .i2(B),
    .o1(WIRE_3));

or_2  b2v_inst9(
    .i1(WIRE_2),
    .i2(WIRE_3),
    .o1(Co));

endmodule
```

- Explicit, easy for programs (EDA tools) reading & processing
- Needs to learn a new computer language (HDL) first
- Easy to scale up for complex designs
- **Higher level logic design** (ex. RTlevel)

Better ask twice than lose you wayonce.

Q&A