Switching Circuit
& Logic Design

Basic Logic Design Practicing
- Quartus II Schematic

Switching Circuit
& Logic Design

Logic Design & Simulation Practicing

--Introduction to Quartus II Schematic Tools

Lecturer: TA 謝明倫 (BL-421)
yans@media.ee.ntu.edu.tw

交換電路與邏輯設計課程 四班共同教學
Professors: 吳安宇 簡韶逸 盧奕璋 江介宏

臺灣大學

Outline

- Introduction
  - design flow
- Access Quartus
  - Workstation
    - workstation list
    - Moba connection
    - Launch Quartus
    - Basic commands
  - Install it yourself
- Quick Trial
  - Create project
  - Creat bdf
  - Compile
  - Simulate vwf
  - Save project
- Design flow
  - 1 Divide system
  - 2 Truth table
  - 3 K-map
  - 4 Boolean expression
  - 5 Pre-draw
  - 6 Quartus design
    - about Lab1
    - FullAdder1.bdf
    - FullAdder1.bsf , module
    - FullAdder4.bdf , array of pins
    - compile
  - 7 Verify (simulation)
- Homework
- Verilog HDL introduction
- Summary
- Q&A and Ref. , Useful tools

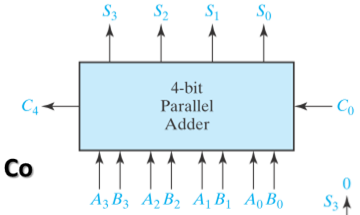
Introduction

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 detailedly 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 it first - 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

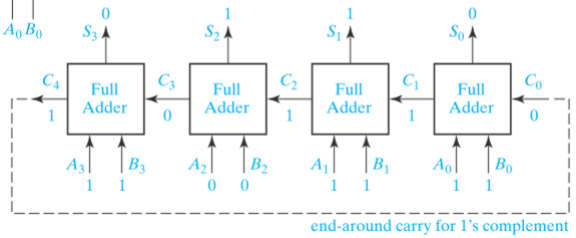
Comb. Ckt Design flow

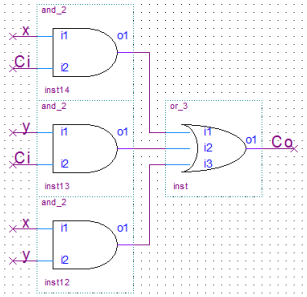
- Gate-level schematic design flow:
  1. Divide system
  2. Truth table
  3. K-map
  4. Boolean expression
  5. Draw it first
  6. Design with tools
  7. Verify & debug



Co

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

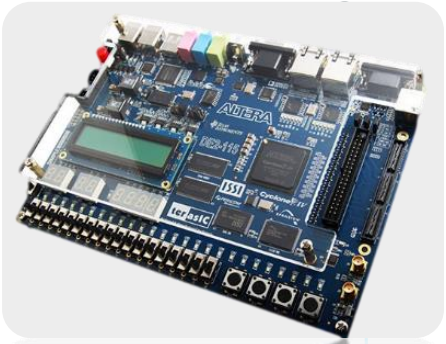




What is Quartus II ?

- It's a software tool for computer-aided digital circuit design
  - produced 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

One must have good tools in order to do a good job.

Access Quartus II

You can choose…

- Connect to workstation of our department
  - Already installed and well setup
  - You need to install a “SSH clients software”

- Install the software yourself on your device
  - If you choose this, please install the version 13.1

Connect to workstation

Workstations

- Why workstations?
  - Multiple Users, multiple tasking
  - Stable Operations
- How to run tasks on the workstations?
  - Operating System: Unix-like
    - Example: Linux
    - Example: Solaris
  - User Interface
    - Text Mode
    - X-Window


```
Login: r02024
r02024@140.112.20.85's password:
Last login: Sun Oct 26 23:14:12 2014 from media6.ee.ntu.edu.tw
[r02024@cad42 ~]$ echo Hello World
Hello World
[r02024@cad42 ~]$
```

Lab 231 workstation list

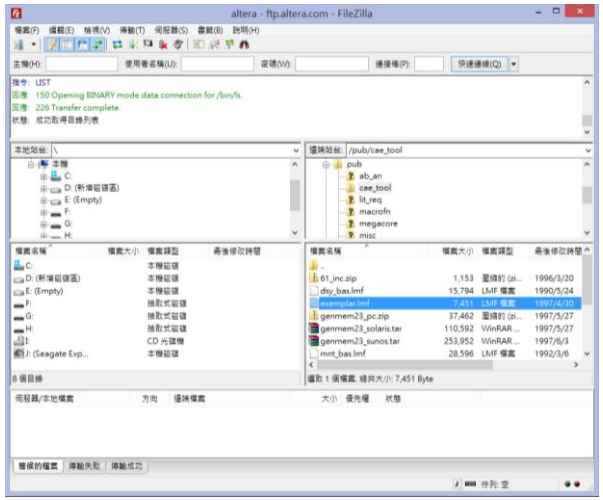
- http://cad.ee.ntu.edu.tw/ws_list.htm
- 140.112.20.??

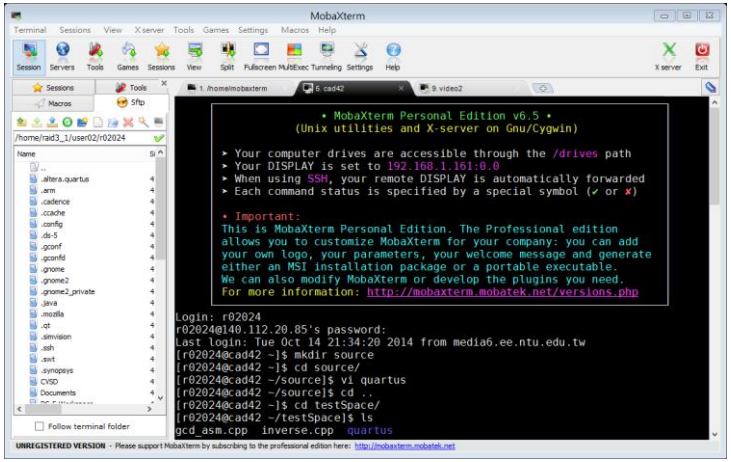
IP	Name	Type	CPU	CPU Clock	Memory	OS
.59	cad16	IBM X3400	Intel Xeon 64	2.4 GHz * 16	100 G	RHEL 5
.60	cad17	IBM X3550	Intel Xeon 64	2.4 GHz * 16	20 G	RHEL 5
.61	cad18	SUN Blade 2500	UltraSPARC	1.28 GHz*2	8 G	Solaris 10
.62	cad19	SUN Blade 2000	UltraSPARC	1.2 GHz * 2	8 G	Solaris 10
.63	cad20	SUN Blade 2000	UltraSPARC	1.2 GHz * 2	8 G	Solaris 10
.64	cad21	DELL R620	Intel Xeon 64	2 GHz * 32	48 G	CentOS 5
.65	cad22	SUN Blade 2500	UltraSPARC	1.28 GHz * 2	8 G	Solaris 10
.66	cad23	DELL R620	Intel Xeon 64	2 GHz * 16	96 G	CentOS 5
.67	cad24	SUN Fire 280R	UltraSPARC	1.2 GHz * 2	4 G	Solaris 9
.68	cad25	SUN Fire 280R	UltraSPARC	1.2 GHz * 2	4 G	Solaris 9
.69	cad26	SUN Fire 280R	UltraSPARC	1.2 GHz * 2	4 G	Solaris 9
.70	cad27	IBM x260	Intel Xeon 64	3.2 GHz * 4	8 G	RHEL 4
.71	cad28	IBM x260	Intel Xeon 64	3.2 GHz * 4	8 G	RHEL 4
.72	cad29	IBM e336	Intel Xeon 64	3.2 GHz	5 G	RHEL 4
.73	cad30	IBM X3650	Intel Xeon 64	2 GHz * 16	12 G	SUSE 11
.74	cad31	DELL R620	Intel Xeon 64	2.2 GHz * 32	48 G	CentOS 6
.75	cad32	SUN Blade 2000	UltraSPARC	1.015 GHz * 2	8 G	Solaris 8
.76	cad33	IBM X3500	Intel Xeon 64	2 GHz * 16	20 G	RHEL 4
.77	cad34	IBM X3650	Intel Xeon 64	2.4 GHz * 8	12 G	RHEL 5
.78	cad35	SUN V20z	AMD Opteron	2.4 GHz * 2	4 G	RHEL 4
.79	cad36	SUN V20z	AMD Opteron	2.4 GHz * 2	4 G	RHEL 4
.80	cad37	SUN V20z	AMD Opteron	2.4 GHz * 2	4 G	RHEL 4
.81	cad38	IBM X3650	Intel Xeon 64	2.4 GHz * 16	96 G	RHEL 5
.82	cad39	IBM X3650	Intel Xeon 64	2 GHz * 24	40 G	CentOS 5
.83	cad40	IBM X3550	Intel Xeon 64	2 GHz * 24	8 G	CentOS 5
.84	cad41	IBM X3550	Intel Xeon 64	2.1 GHz * 24	64 G	CentOS 6
.85	cad42	IBM X3500	Intel Xeon 64	2 GHz * 24	32 G	CentOS 5

Quartus II is on these workstations

Tools you need

- FTP client
  - Famous: FileZilla
- X server
  - Famous: Xming
- SSH client
  - Famous: Putty (contain Xming)
- 3 in 1: MobaXterm (SFTP+SSH+X server)





MobaXterm

MobaXterm (1/4) Download

1

MobaXterm - the complete toolbox for remote computing

2

Features	Personal Edition	Professional Edition
Installation/Uninstallation support	✔	✔
X server support	✔	✔
Remote computing clients	✔	✔
Advanced options	✔	✔
	Free download	PayPal VISA BANK Subscribe now

3

MobaXterm download

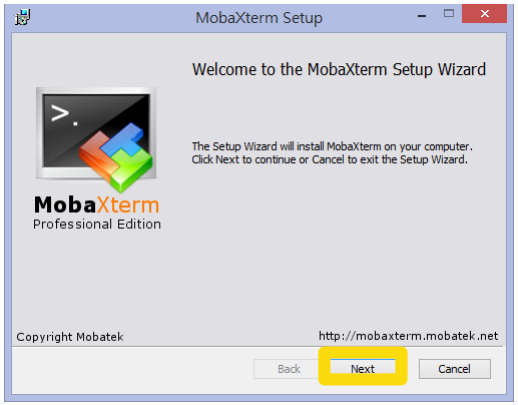
Download MobaXterm version 7.3 for free

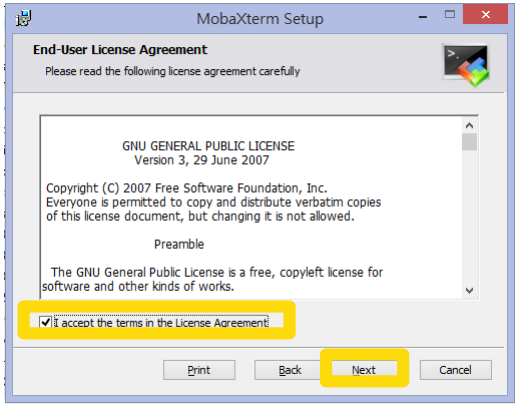
- MobaXterm v7.3 (installer)
- MobaXterm v7.3 (portable edition)
- MobaXterm plugins

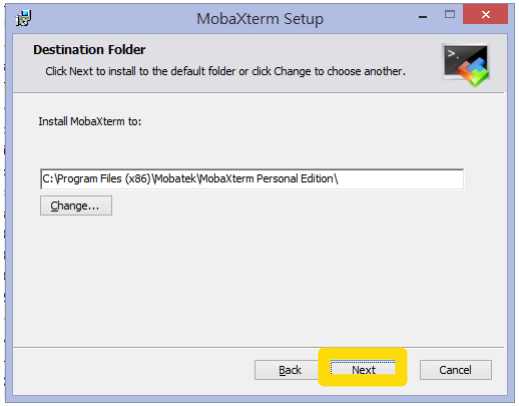
If admin O
If admin X

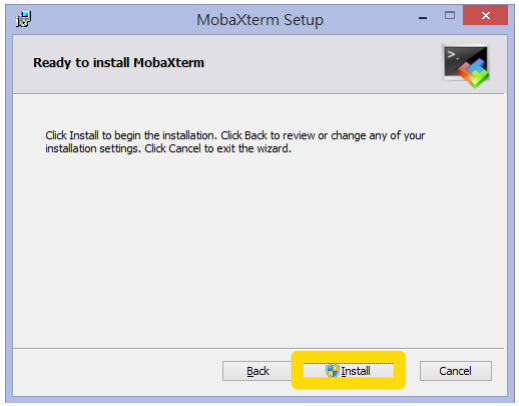
- Download MobaXterm and third party sources

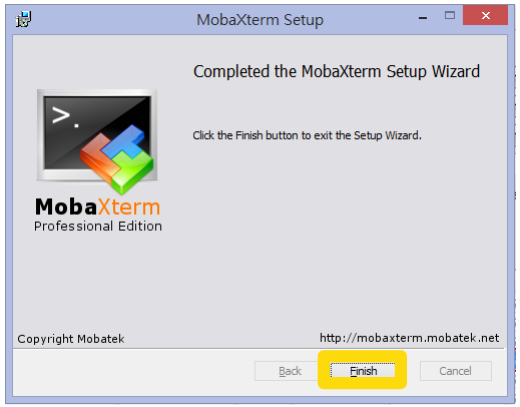
MobaXterm (2/4) Installation



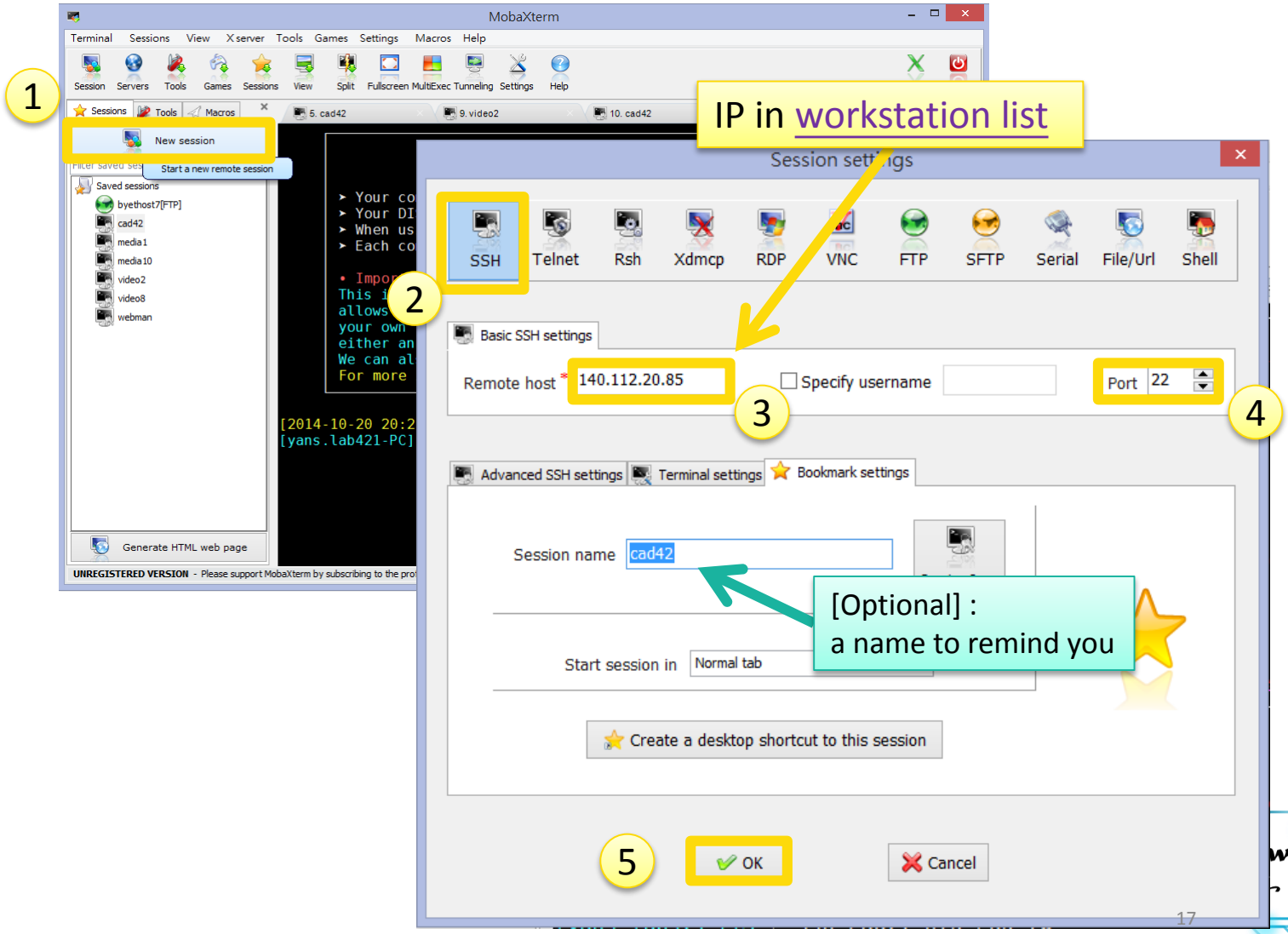






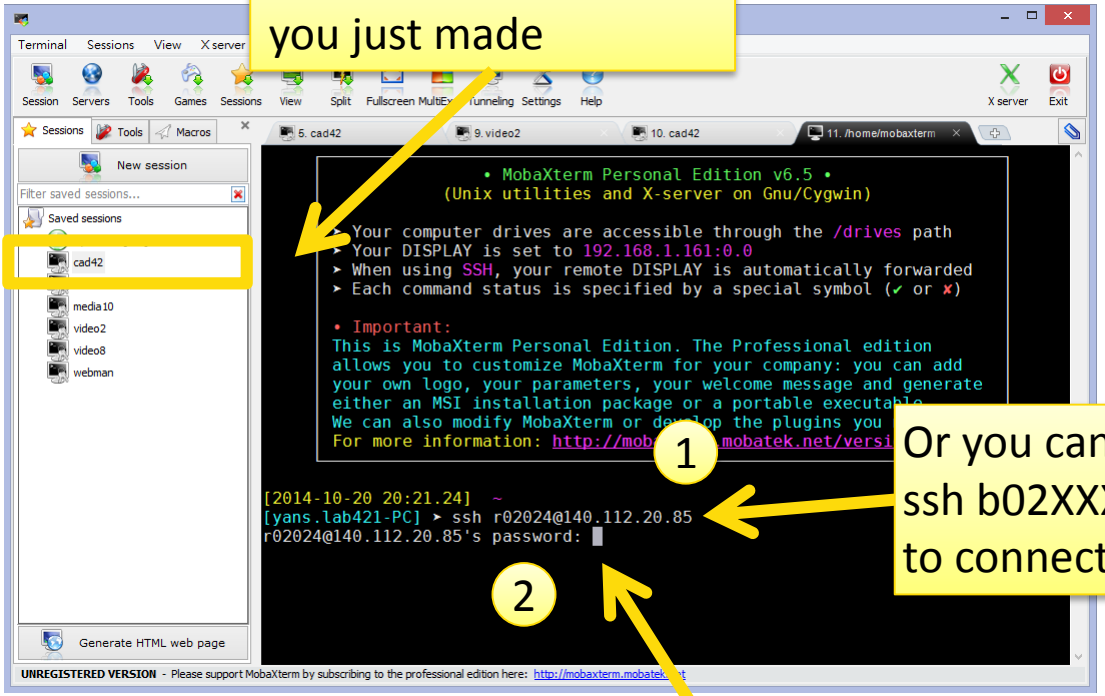


MobaXterm (3/4) Setup



witching Circuit
Logic Design

MobaXterm (4/4) Connection




```
Login: r02024
r02024@140.112.20.85's password:
Last login: Sat Oct 18 15:54:11 2014 from media6.ee.ntu.edu.tw
[r02024@cad42 ~]$
```

Your Account

- 帳號命名規則：
  - 學號之「系所代碼」為 943 與 901 同學：
    將學號中「系所代碼」去掉，即為帳號。
    - 例：學號為「b02901000」→帳號為「b02000」
  - 其它系所代碼同學，將學號中英文字母後
    第一個數字去掉即為帳號。
    - 例：學號為「r02942000」→帳號為「r2942000」
  - Please check if there is any log-in problem ?

Rules of Workstations

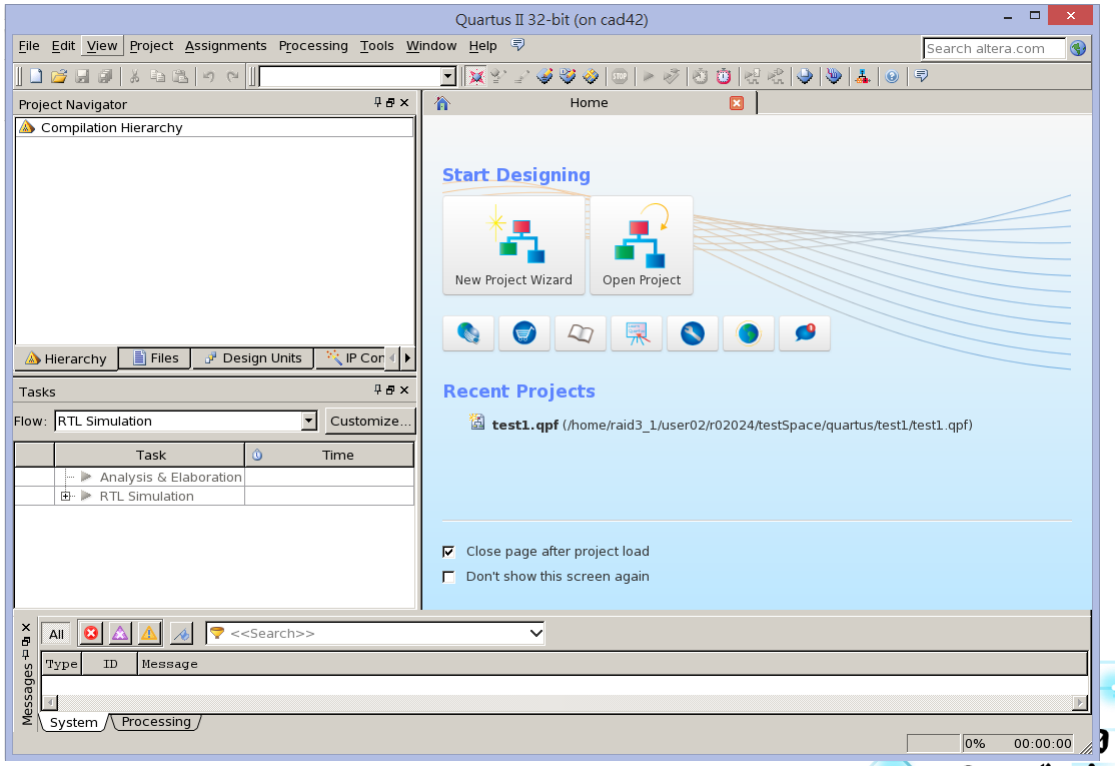
- 實驗室規則：
  - 1. 請勿任意 reboot 主機，破壞系統或做出對系統有害之行為，或將帳號借予他人使用，否則若經查獲，立即刪除帳號，並交與教授處理。
  - 2. 硬碟使用空間以大學部 1G、碩士班 5G、博士班 10G 為限，若需超過，請填寫「系統維護申請表」，申請更改 QUOTA。
  - 3. 請愛護使用實驗室儀器設備，保持桌面整潔，座椅用畢歸位，垃圾自行帶走。
  - 4. 借閱本實驗室之軟體、書籍、使用手冊時，請按時歸還。
  - 5. 個人資料請自行備份，並於畢業時將個人目錄下的檔案清理乾淨。
  - 6. 大學部同學帳號預設期限為一學期，碩士班同學為兩年，博士班同學為四年，若需要延長，請填寫系統維護申請表。
  - 7. 其它注意事項請參閱本實驗室內的實驗室公布欄與實驗室網頁 http://cad.ee.ntu.edu.tw。

Launch Quartus II

- type:

```
[r02024@cad42 ~]$ tcsh
[r02024@cad42 ~]$ source /home/raid5_4/raid2_3/solaris/Quartus/.cshrc
[r02024@cad42 ~]$ quartus&
```

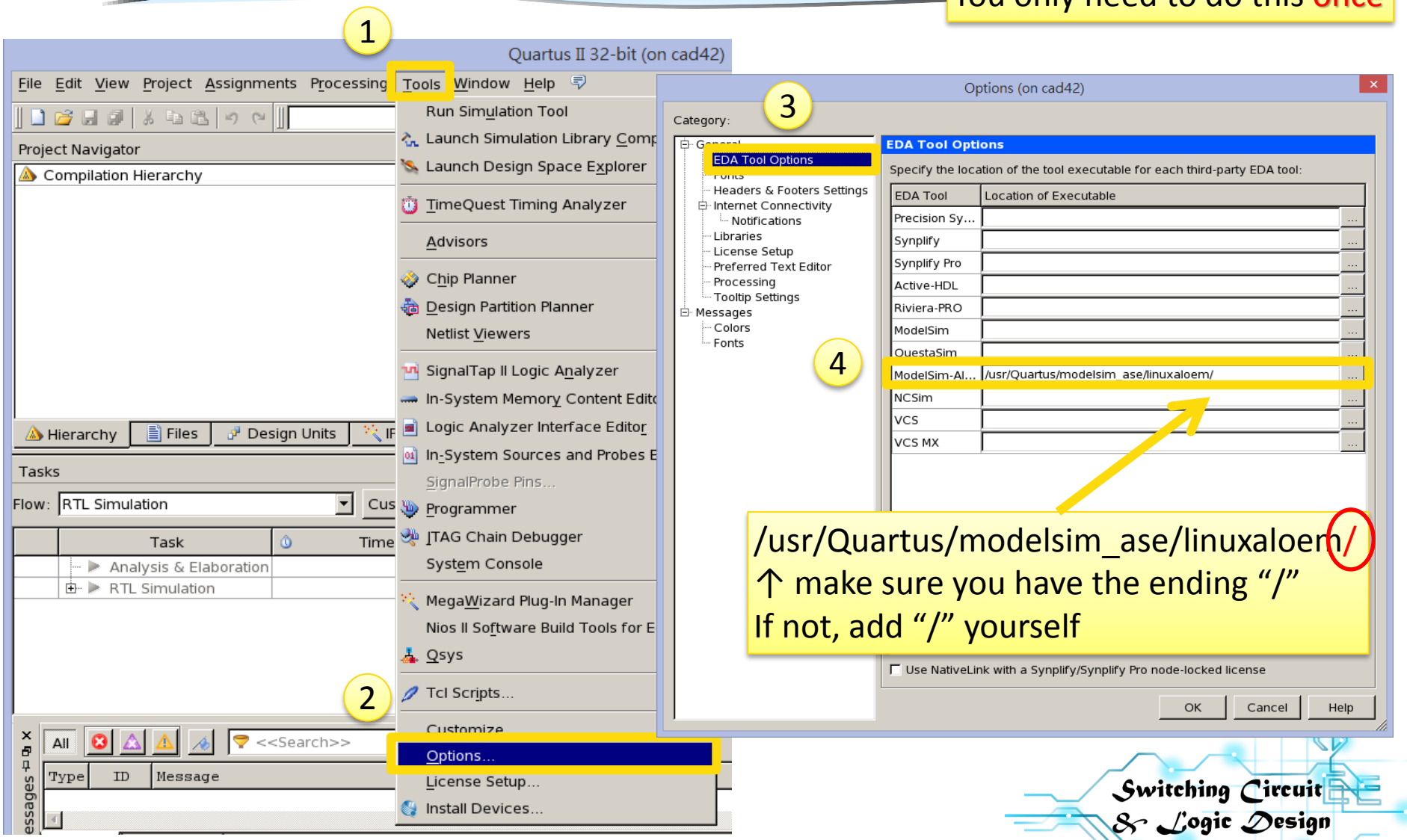
```
tcsh
source /home/raid5_4/raid2_3/solaris/Quartus/.cshrc
quartus &
```



[Hint] You can click middle mouse button as “paste” on Moba

Check the Settings of Quartus

You only need to do this once



Basic Commands (1/4)

- Files
  - `ls` : lists your files
  - `ls -l` : lists your files in long format
  - `ls -a` : lists all files, including the ones whose filename beginning in a dot
  - `cp` *src-filename dest-filename* : copy files
  - `mv` *src-filename dest-filename* : move files (can rename files)
  - `rm` *filename* : delete this file
    - Add "`-r`" for directories. ex. `cp -r`
  - `more` *filename* : show the content of the file

Basic Commands (2/4)

- Directories
  - `mkdir` *dirname* : make a new directory
  - `cd` *dirname* : change directory
  - `cd ..` : go up on directory level from here
  - `cd ~` : go to user's home directory
  - `rm -rf` *dirname* : remove a directory
  - `pwd` : display your current path

- Compress and decompress files
  - `tar -zcvf` *filename.tgz files* : pack & compress files as filename
  - `tar -zxvf` *filename.tgz* : decompress & unpack filename.tgz

Basic Commands (3/4)

- Processes
  - `ctrl + c` [keyboard]: kill foreground process
  - `ps` : report processes and their pid numbers
  - `kill` *pid#* : kill the process with the pid#
- Logout
  - `exit`
- More options about command
  - `man` *command-name*: display on-line manual pages

Basic Commands (4/4)

- Reference
  - Chinese
    - http://boson4.phys.tku.edu.tw/UNIX/Unix%20Command/index_basic.htm
    - http://www.lib.cgu.edu.tw/instruction/basiccmd.html
  - English
    - http://mally.stanford.edu/~sr/computing/basic-unix.html
    - http://www.math.utah.edu/lab/unix/unix-commands.html
    - http://www.computerhope.com/unix/overview.htm

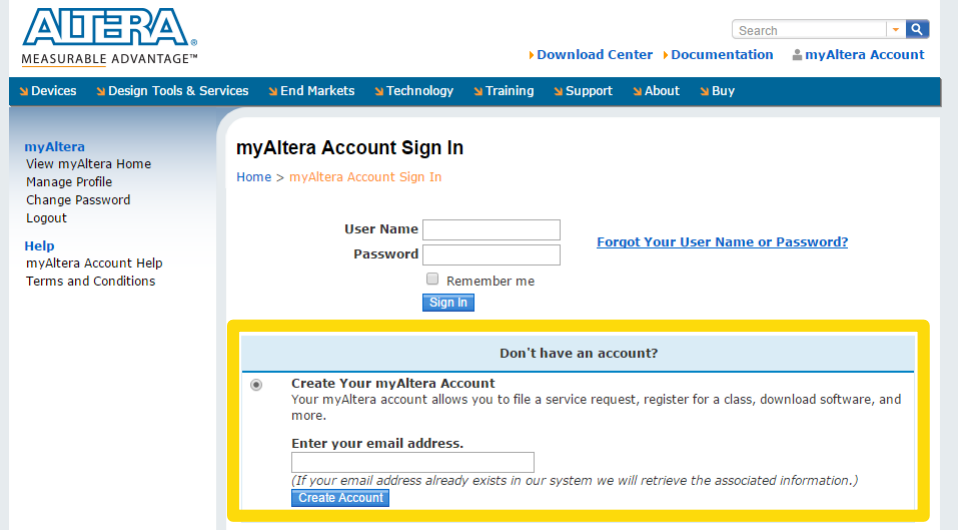
Not recommended

Install it yourself

Install

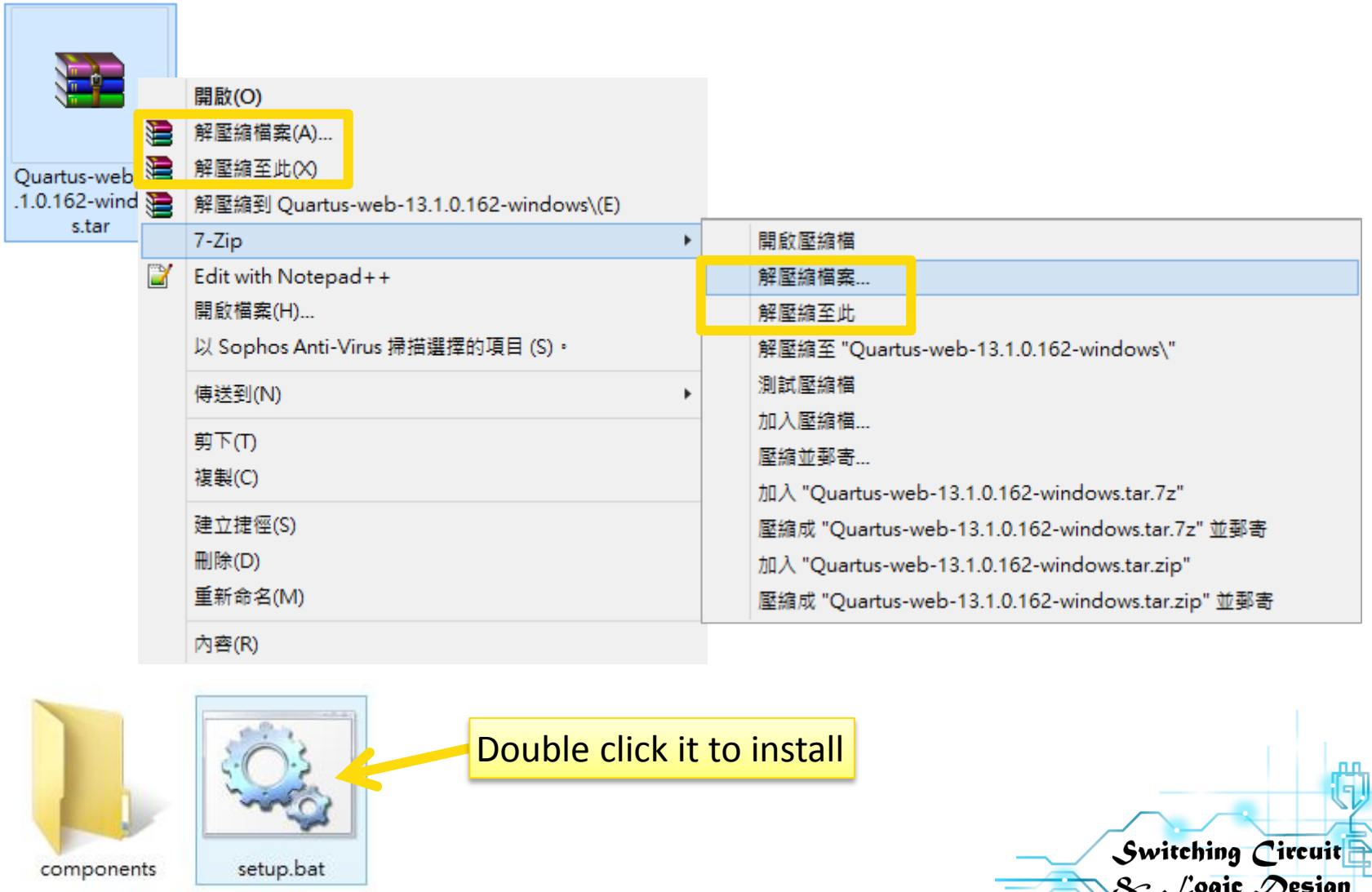
- Altera Quartus Website
- Download link
  - http://dl.altera.com/13.1/?edition=web

- Make sure 13.1 web edition
  - web edition is free, but needs registration first

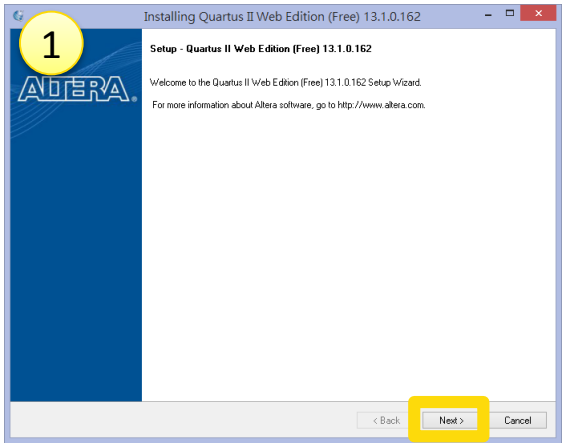


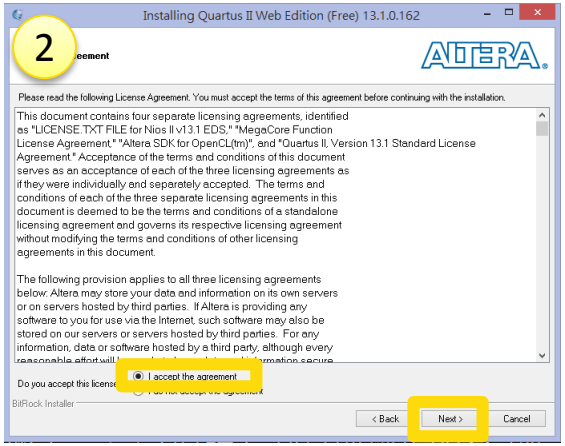
After Registration

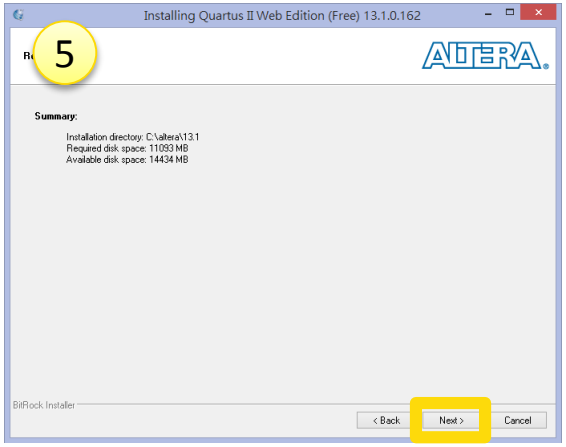
Unpack

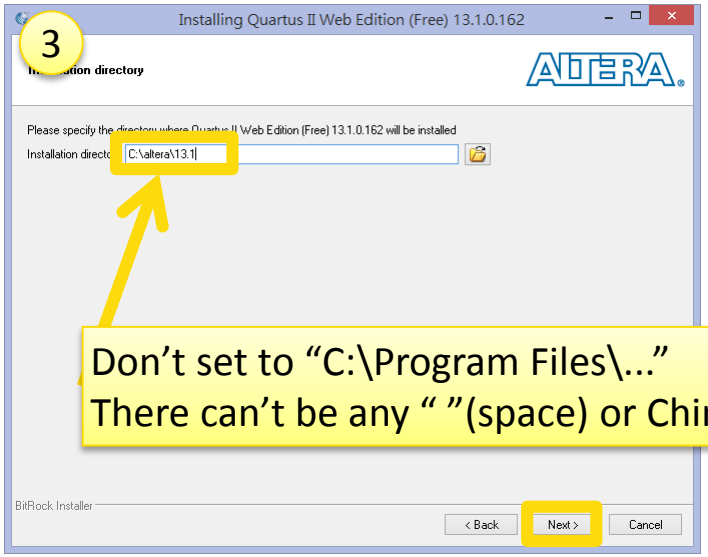

Quartus-web
.1.0.162-wind
s.tar
開啟(O)
解壓縮檔案(A)...
解壓縮至此(X)
解壓縮到 Quartus-web-13.1.0.162-windows\(E)
7-Zip
Edit with Notepad++
開啟檔案(H)...
以 Sophos Anti-Virus 掃描選擇的項目 (S)。
傳送到(N)
剪下(T)
複製(C)
建立捷徑(S)
刪除(D)
重新命名(M)
內容(R)
開啟壓縮檔
解壓縮檔案...
解壓縮至此
解壓縮至 "Quartus-web-13.1.0.162-windows\"
測試壓縮檔
加入壓縮檔...
壓縮並郵寄...
加入 "Quartus-web-13.1.0.162-windows.tar.7z"
壓縮成 "Quartus-web-13.1.0.162-windows.tar.7z" 並郵寄
加入 "Quartus-web-13.1.0.162-windows.tar.zip"
壓縮成 "Quartus-web-13.1.0.162-windows.tar.zip" 並郵寄
components
setup.bat
Double click it to install

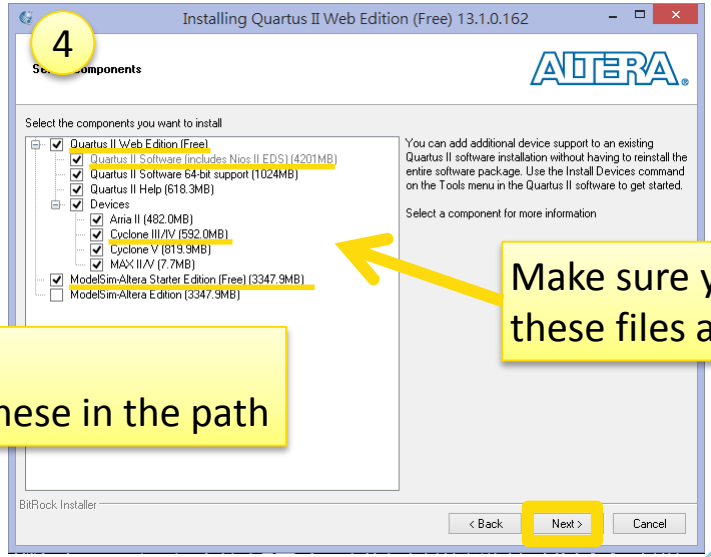
Installation









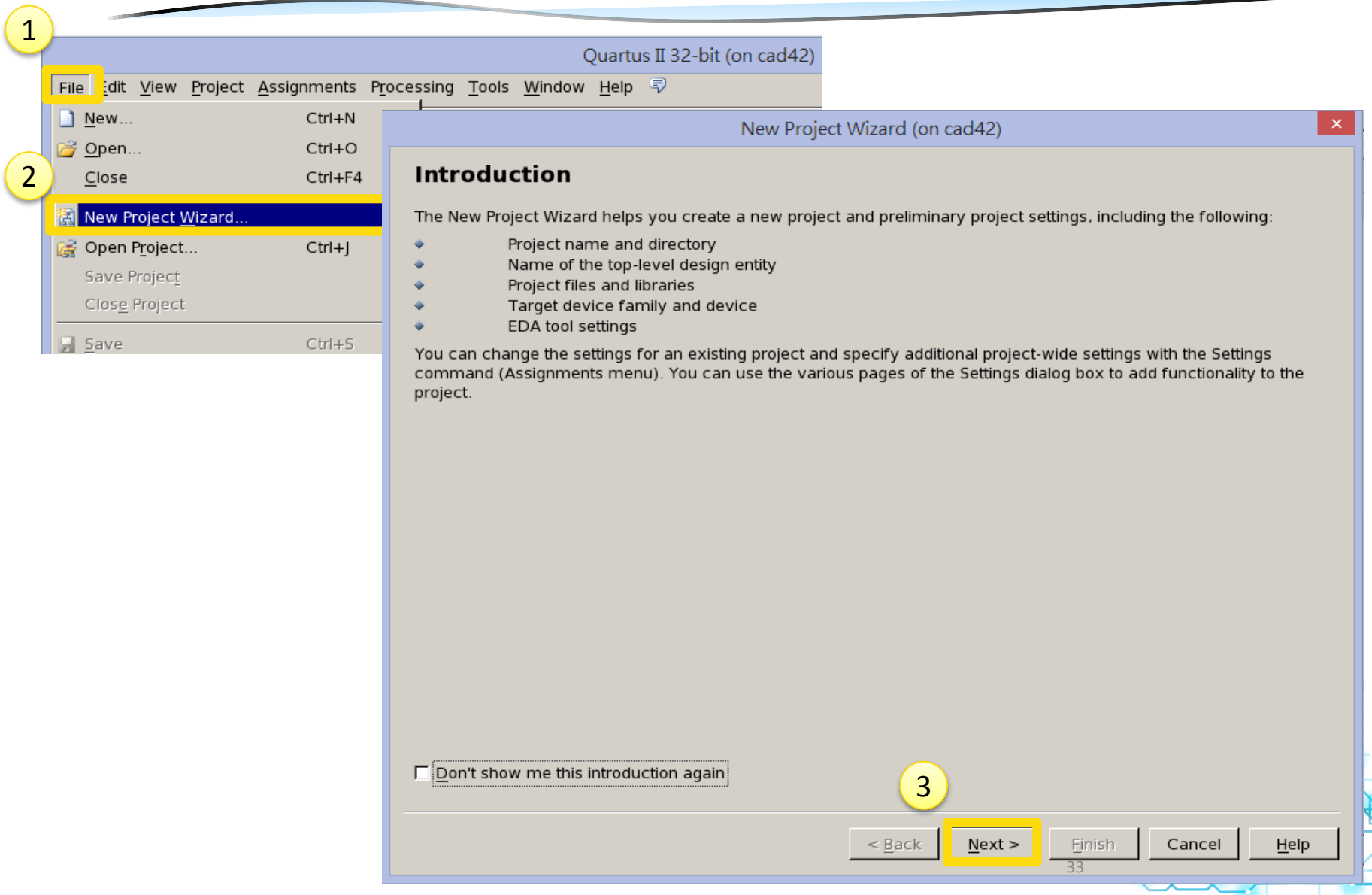


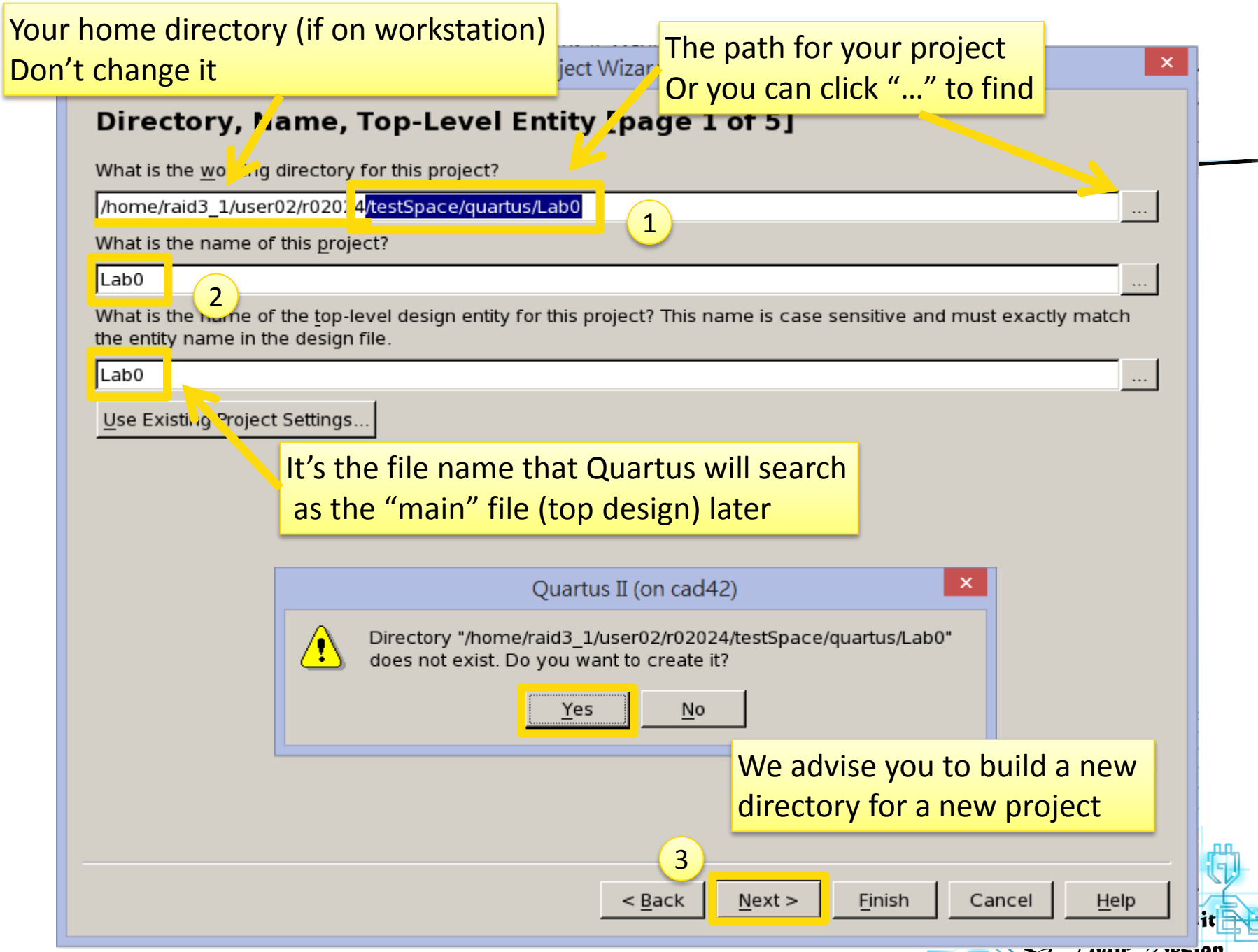
Don’t set to “C:\Program Files\...”
There can’t be any “ ”(space) or Chinese in the path

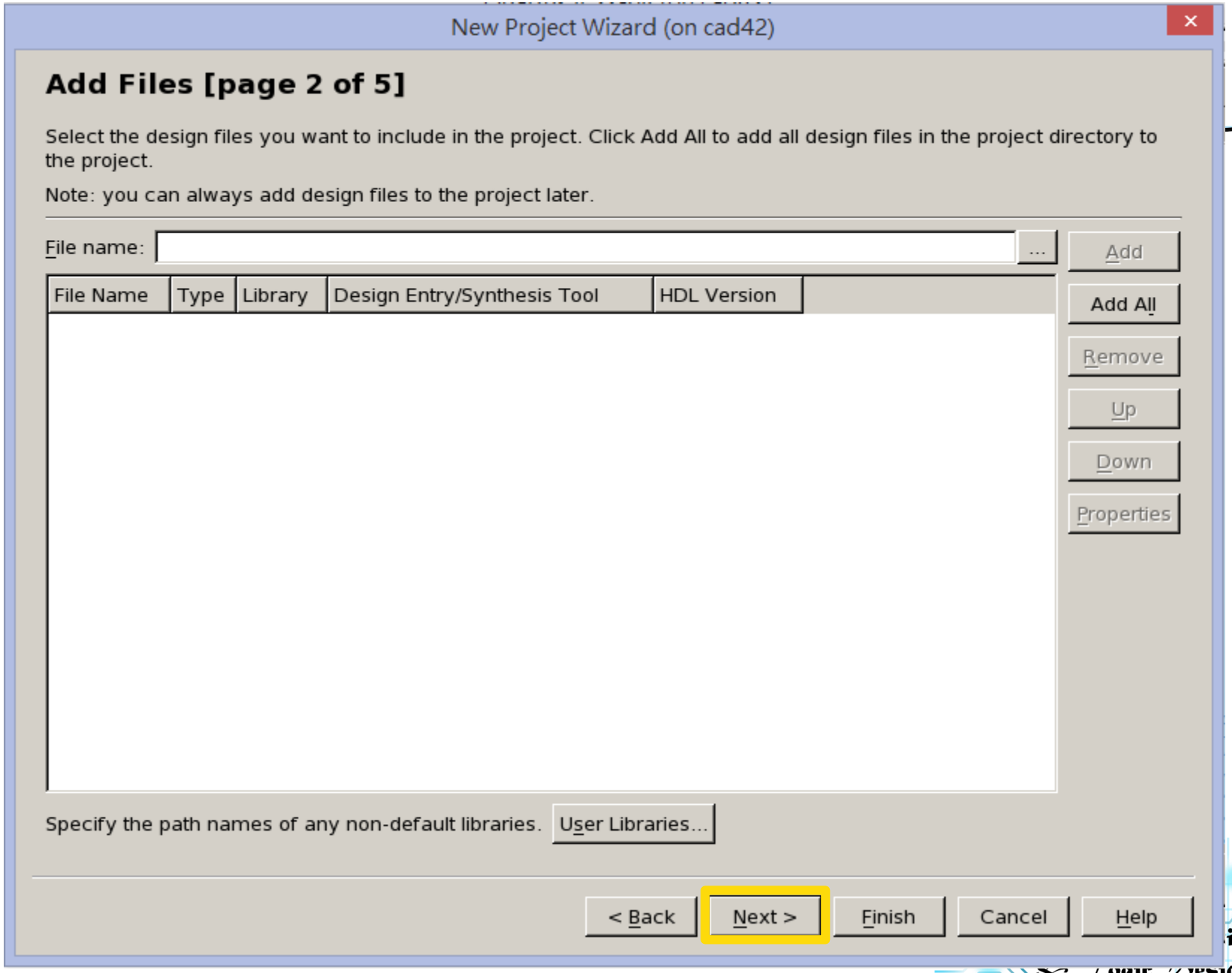
Make sure you have these files at lease

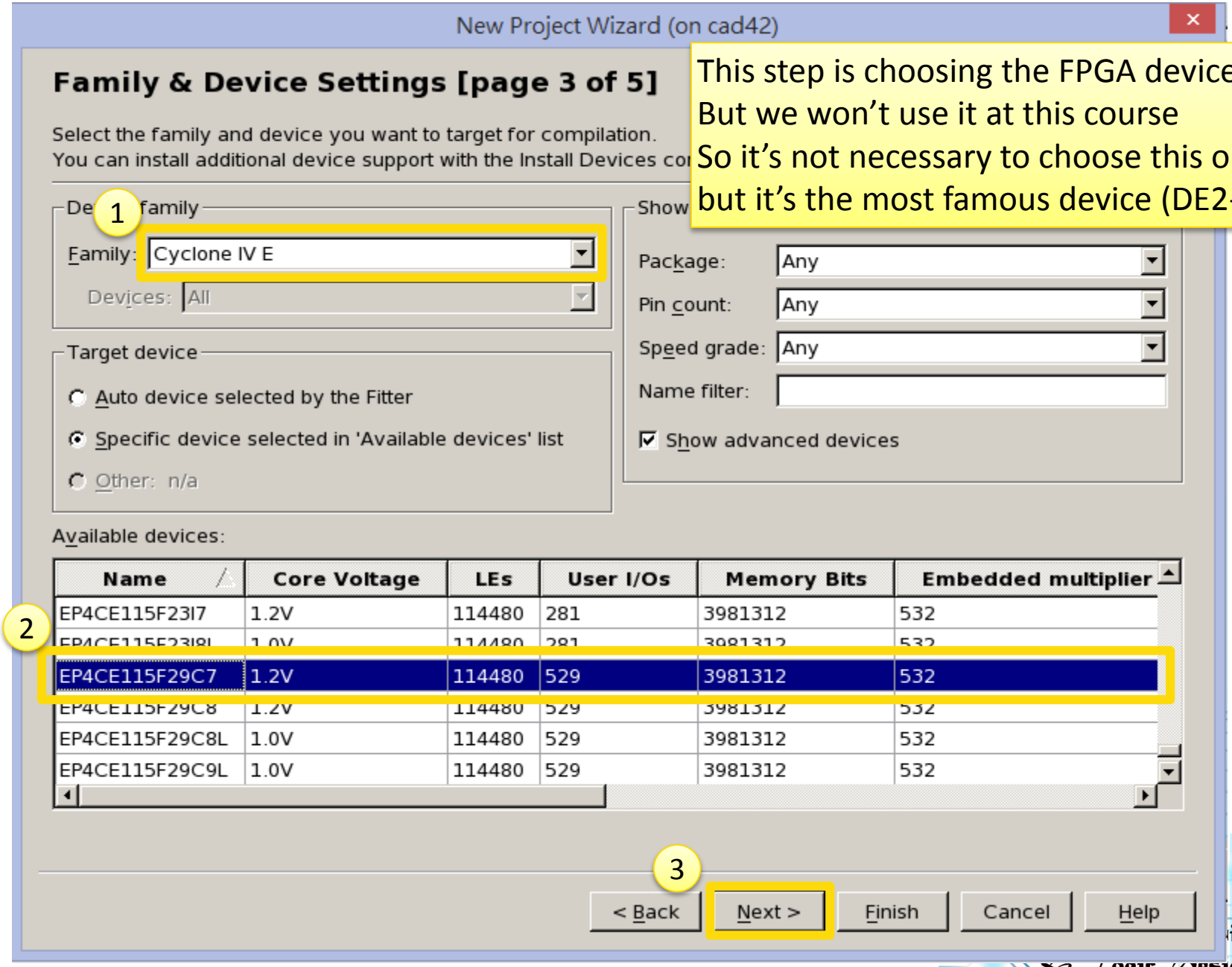
Quartus Schematic Design Trial

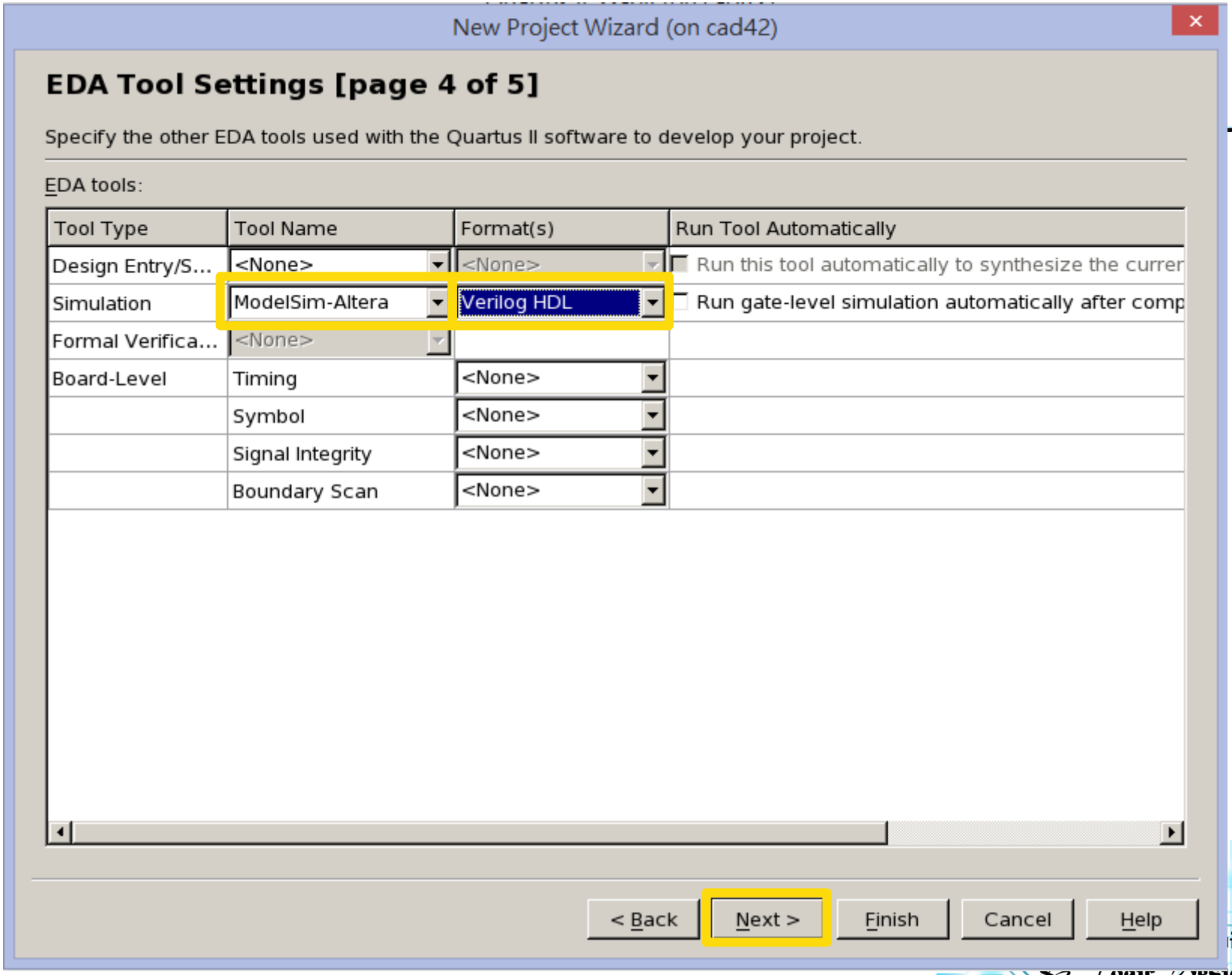
Create a New Project

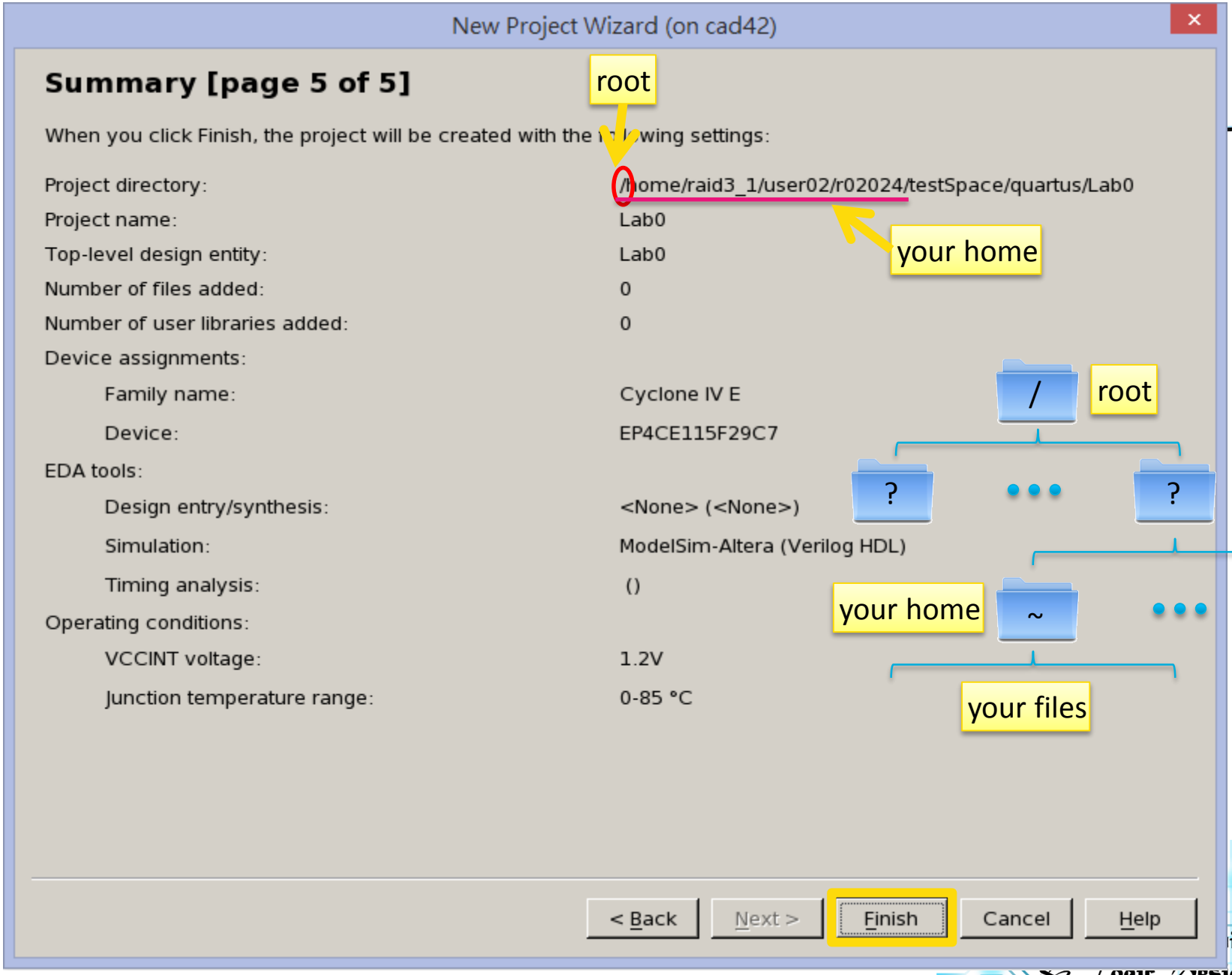



Your home directory (if on workstation)
Don't change it
The path for your project
Or you can click "…" to find
Directory, Name, Top-Level Entity [page 1 of 5]
What is the working directory for this project?
/home/raid3_1/user02/r02024/testSpace/quartus/Lab0
1
What is the name of this project?
Lab0
2
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.
Lab0
Use Existing Project Settings...
It's the file name that Quartus will search as the "main" file (top design) later
Quartus II (on cad42)
Directory "/home/raid3_1/user02/r02024/testSpace/quartus/Lab0" does not exist. Do you want to create it?
Yes
No
We advise you to build a new directory for a new project
3
< Back
Next >
Finish
Cancel
Help

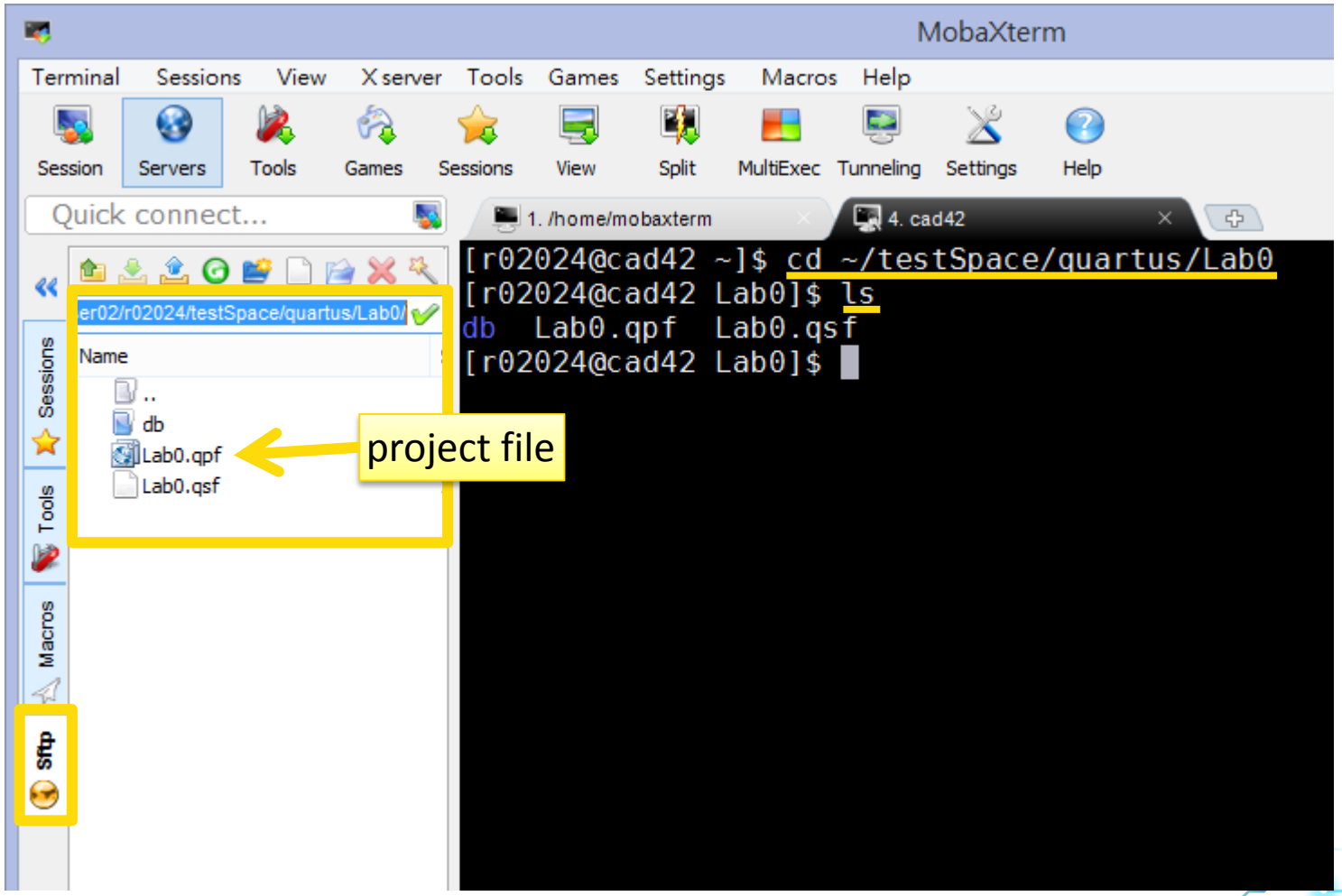

New Project Wizard (on cad42)
Add Files [page 2 of 5]
Select the design files you want to include in the project. Click Add All to add all design files in the project directory to the project.
Note: you can always add design files to the project later.
File name:
...
Add
File Name
Type
Library
Design Entry/Synthesis Tool
HDL Version
Add All
Remove
Up
Down
Properties
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
EP4CE115F29C8 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

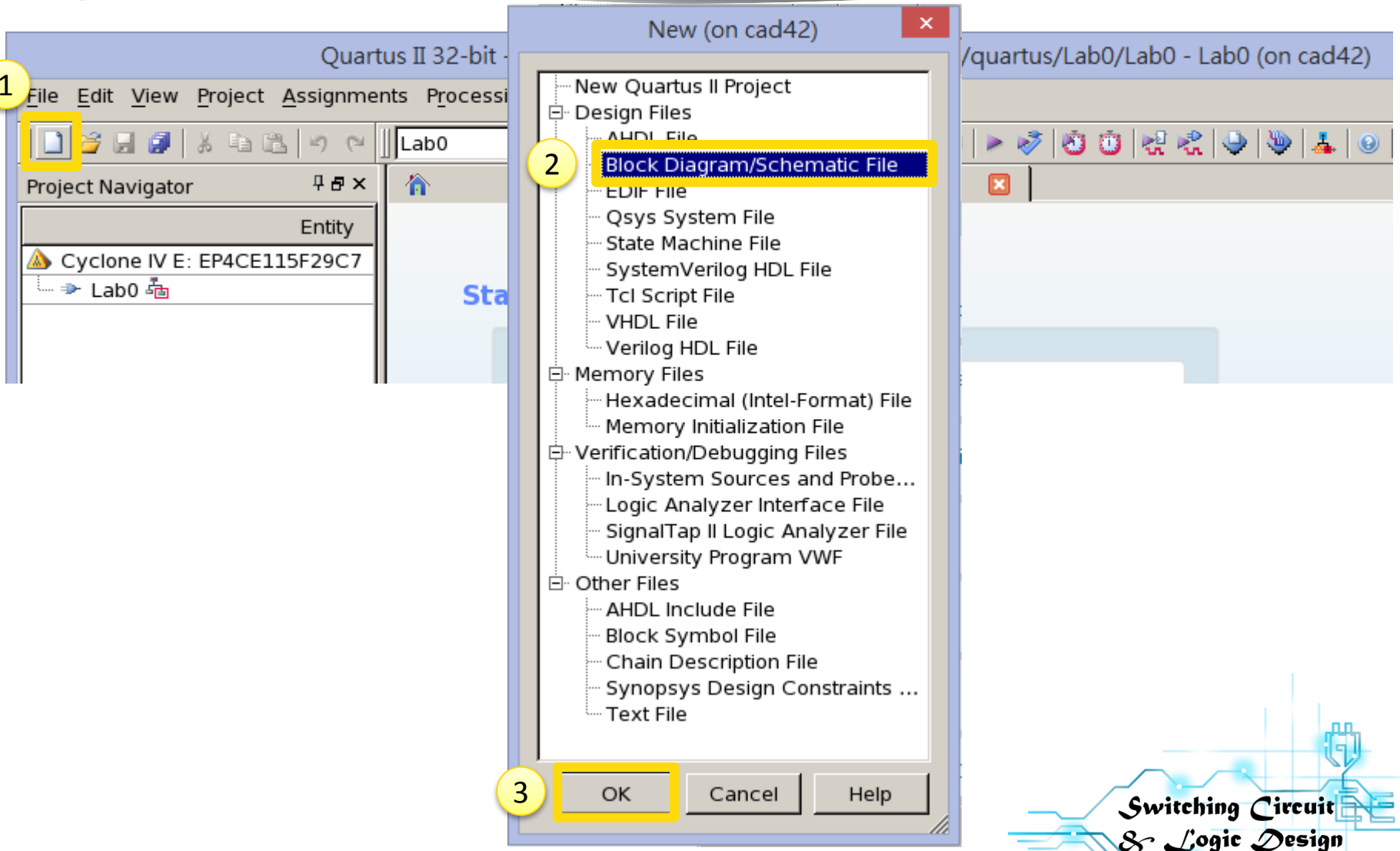

New Project Wizard (on cad42)
EDA Tool Settings [page 4 of 5]
Specify the other EDA tools used with the Quartus II software to develop your project.
EDA tools:
Tool Type
Tool Name
Format(s)
Run Tool Automatically
Design Entry/S...
<None>
<None>
Run this tool automatically to synthesize the currer
Simulation
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 (on cad42)
Summary [page 5 of 5]
root
When you click Finish, the project will be created with the following settings:
Project directory: /home/raid3_1/user02/r02024/testSpace/quartus/Lab0
your home
Project name: Lab0
Top-level design entity: Lab0
Number of files added: 0
Number of user libraries added: 0
Device assignments:
Family name: Cyclone IV E
Device: EP4CE115F29C7
EDA tools:
Design entry/synthesis: <None> (<None>)
Simulation: ModelSim-Altera (Verilog HDL)
Timing analysis: ()
Operating conditions:
VCCINT voltage: 1.2V
Junction temperature range: 0-85 °C
/
root
?
?
your home
~
your files
< Back
Next >
Finish
Cancel
Help

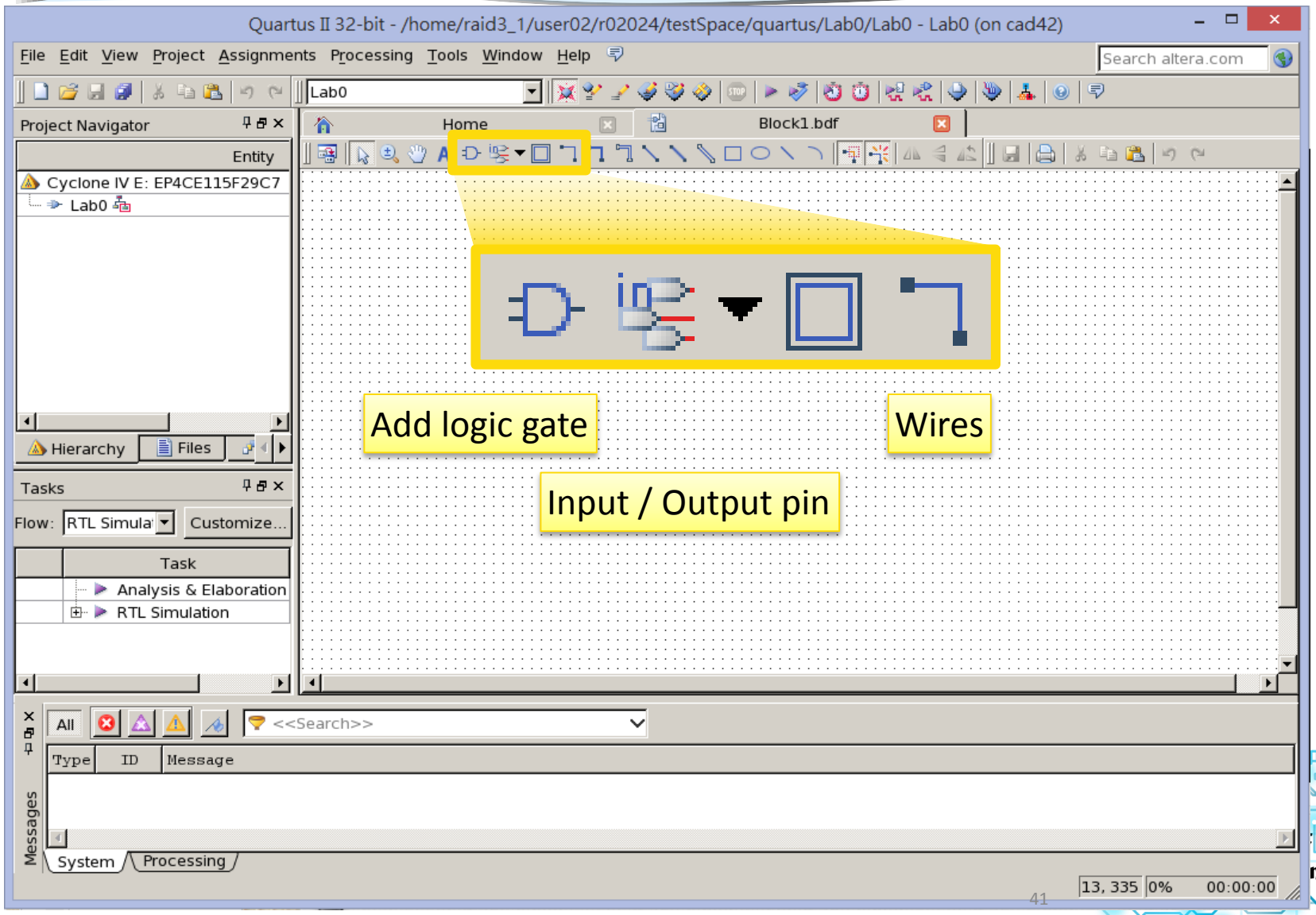
Create a New Project



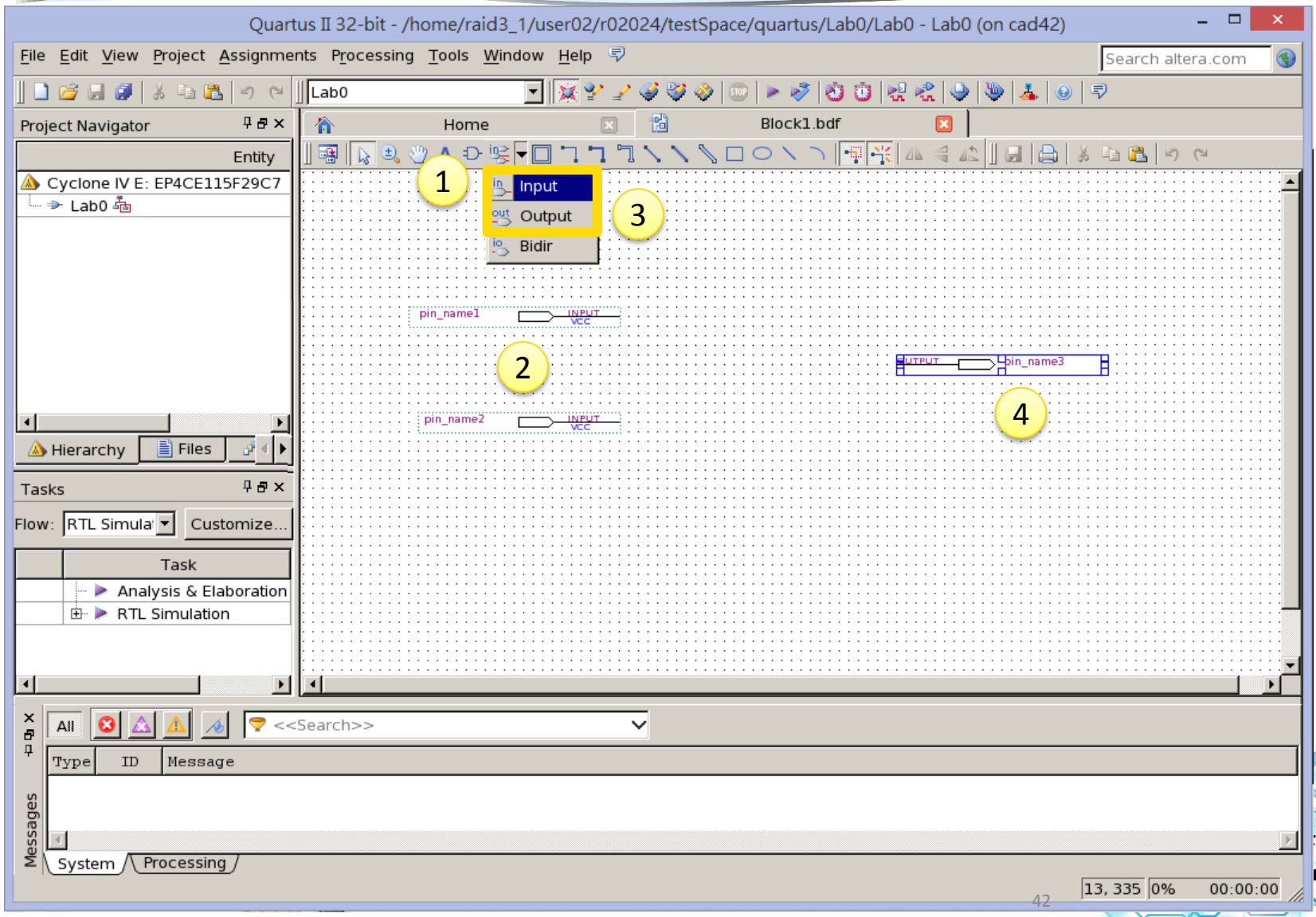
Create a Block Diagram File



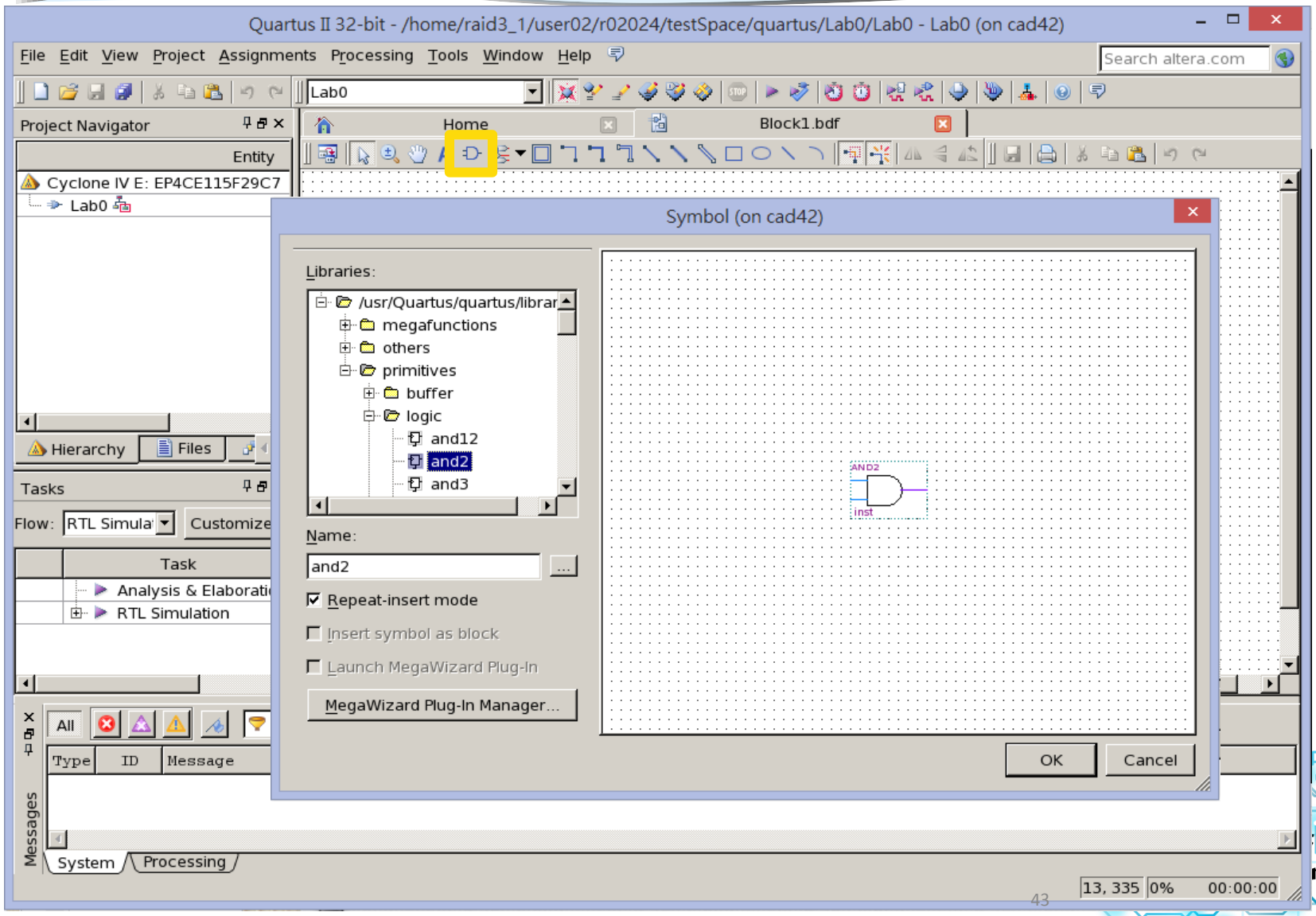
Create a Block Diagram File



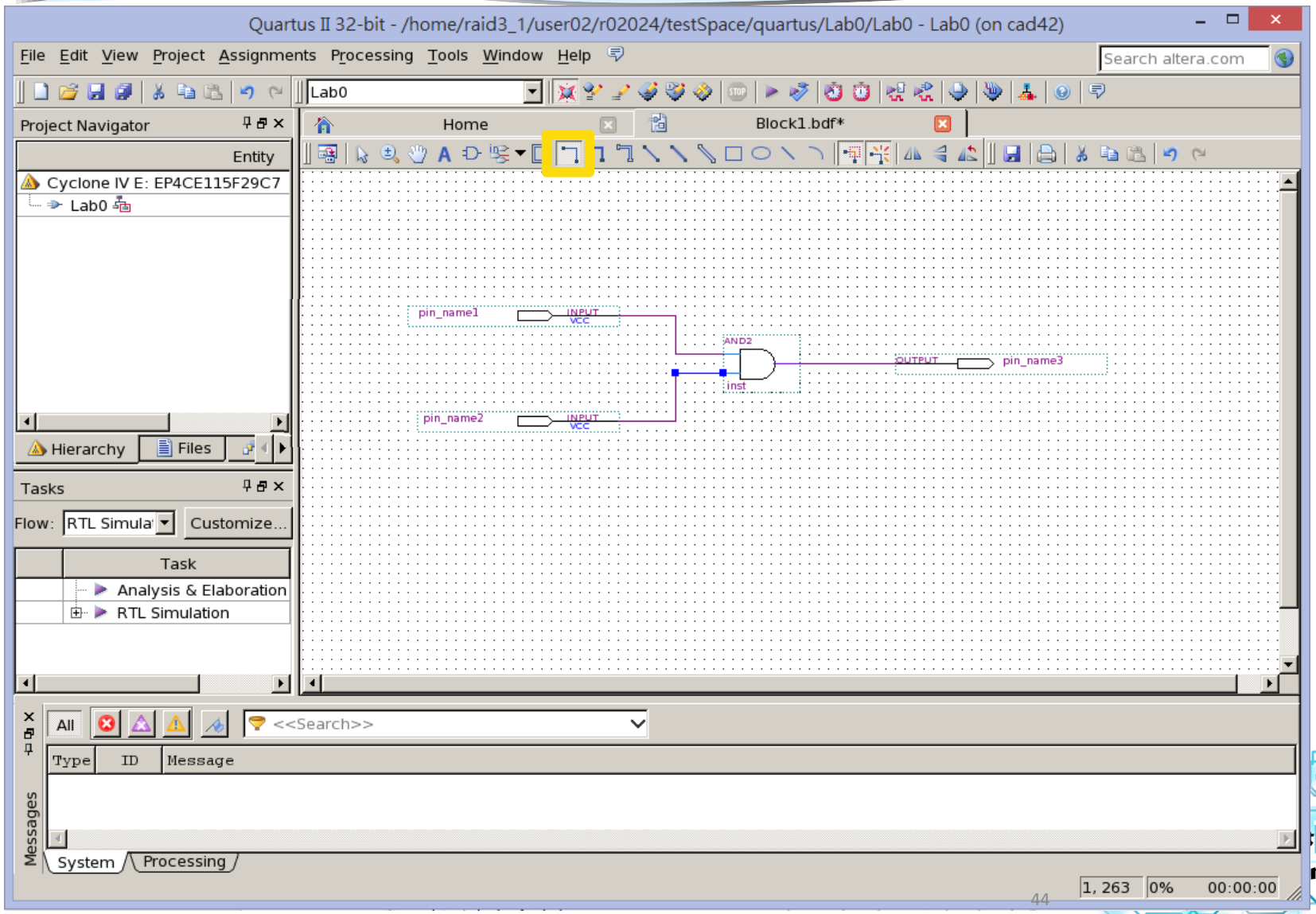
Example


Quartus II 32-bit - /home/raid3_1/user02/r02024/testSpace/quartus/Lab0/Lab0 - Lab0 (on cad42)
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
pin_name2
pin_name3
INPUT
VCC
OUTPUT
All
<<Search>>
Type
ID
Message
Messages
System
Processing
13, 335
0%
00:00:00

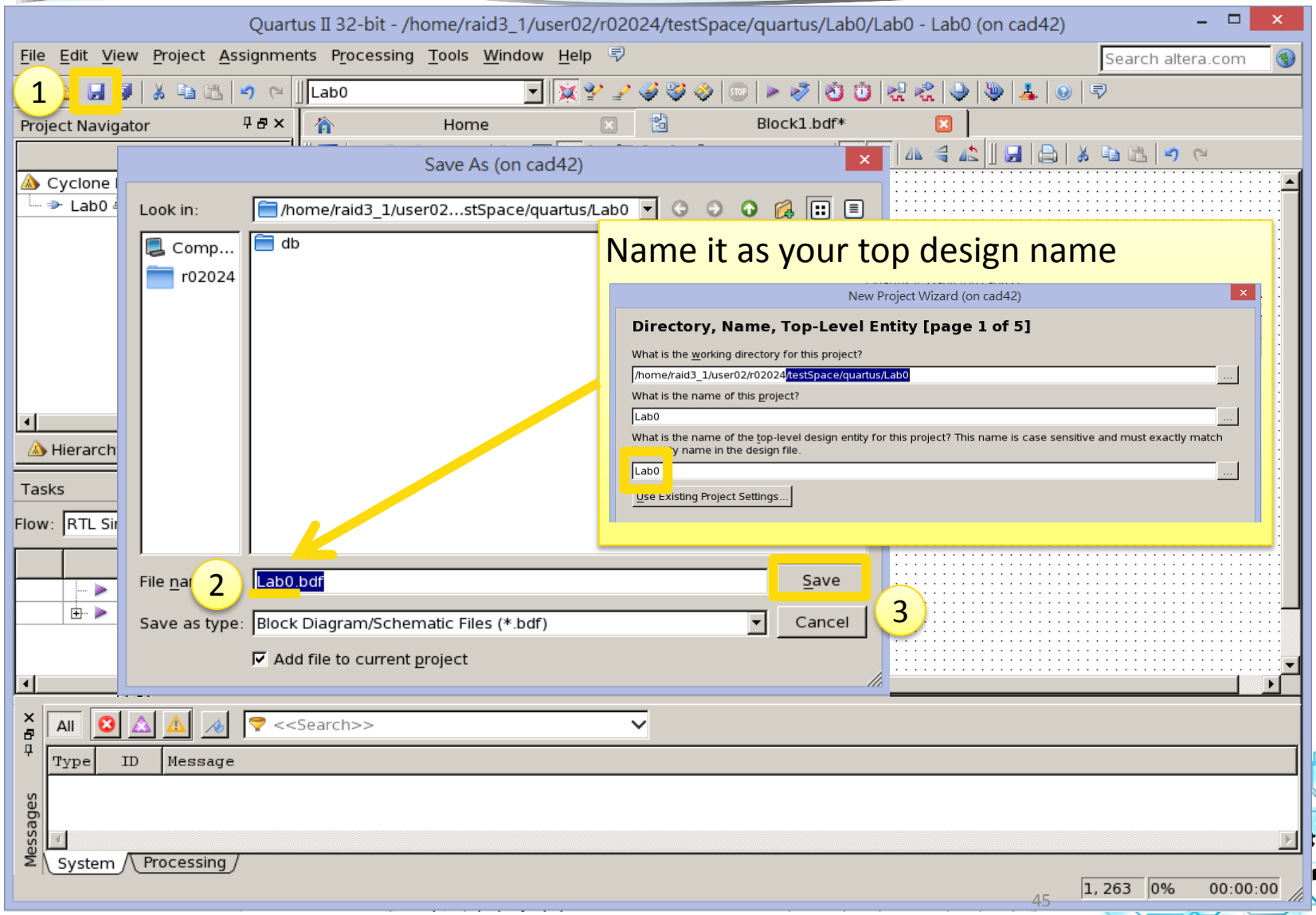
Example


Quartus II 32-bit - /home/raid3_1/user02/r02024/testSpace/quartus/Lab0/Lab0 - Lab0 (on cad42)
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
Lab0
Project Navigator
Entity
Cyclone IV E: EP4CE115F29C7
Lab0
Home
Block1.bdf
Hierarchy
Files
Tasks
Flow: RTL Simula
Customize
Task
Analysis & Elaborati
RTL Simulation
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

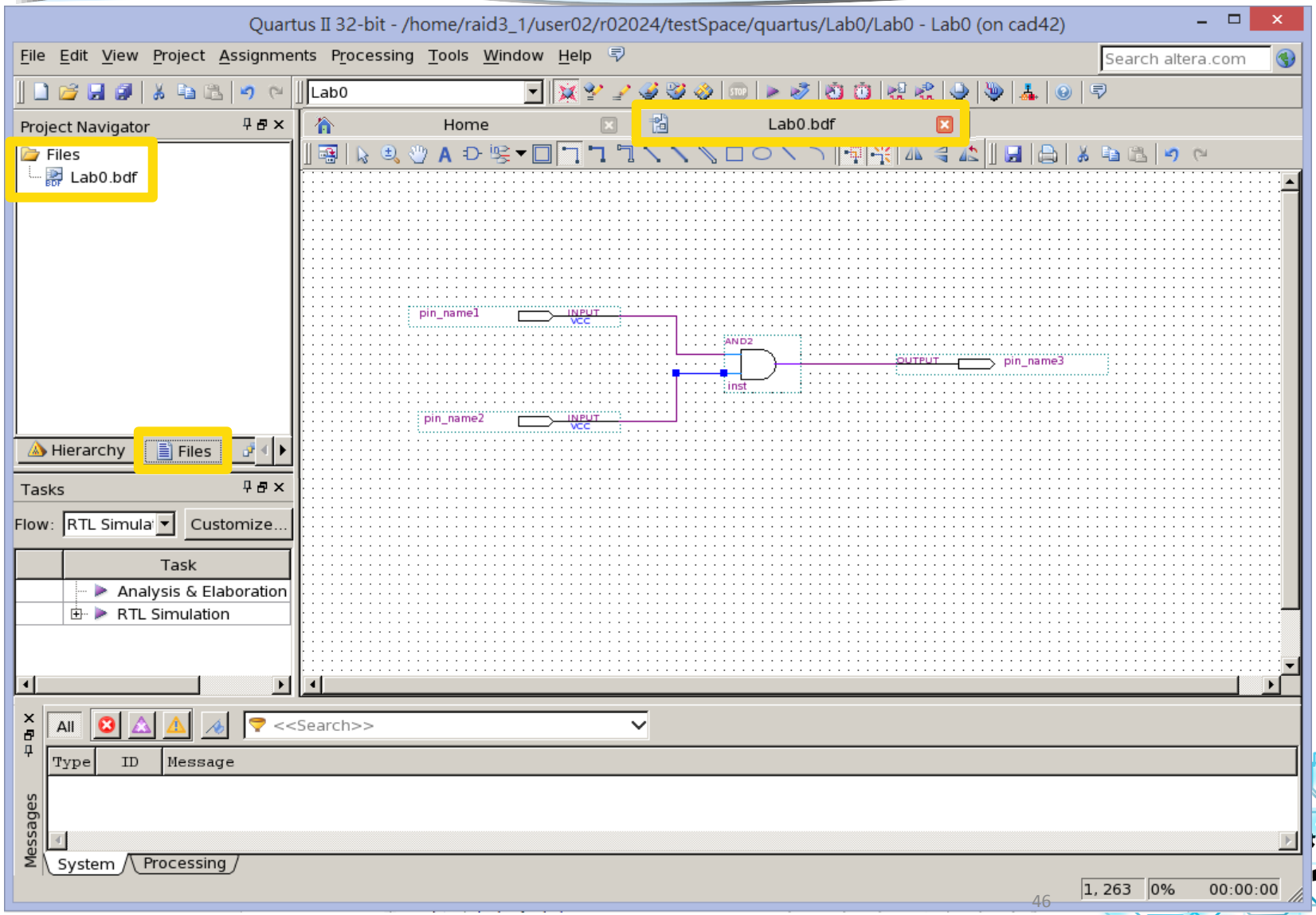
Example


Quartus II 32-bit - /home/raid3_1/user02/r02024/testSpace/quartus/Lab0/Lab0 - Lab0 (on cad42)
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*
pin_name1
INPUT
VCC
pin_name2
INPUT
VCC
AND2
inst
OUTPUT
pin_name3
All
<<Search>>
Type
ID
Message
Messages
System
Processing
1, 263
0%
00:00:00

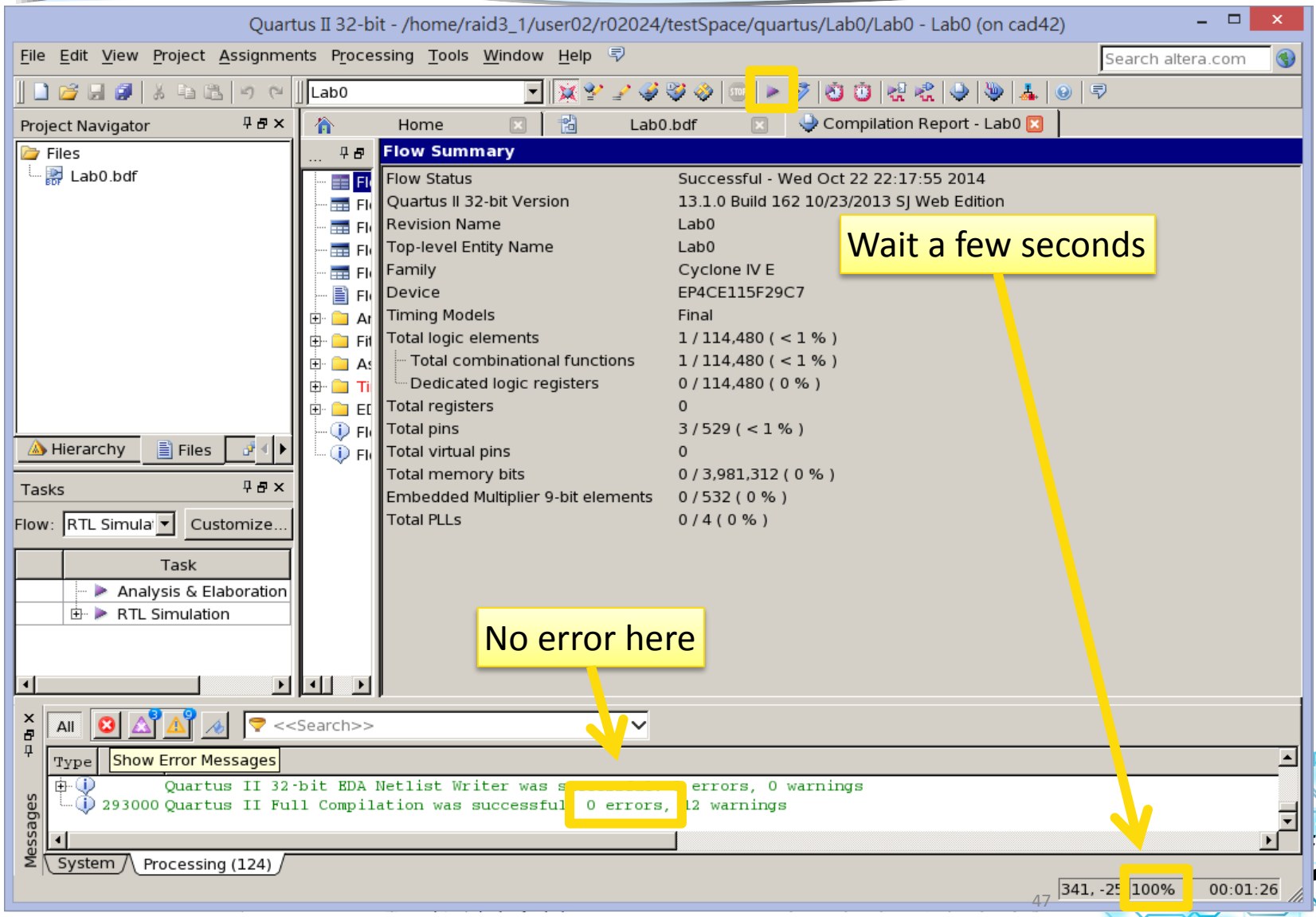
Save it


Quartus II 32-bit - /home/raid3_1/user02/r02024/testSpace/quartus/Lab0/Lab0 - Lab0 (on cad42)
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
1
Lab0
Project Navigator
Home
Block1.bdf*
Save As (on cad42)
Look in:
/home/raid3_1/user02...stSpace/quartus/Lab0
Comp...
r02024
db
Name it as your top design name
New Project Wizard (on cad42)
Directory, Name, Top-Level Entity [page 1 of 5]
What is the working directory for this project?
/home/raid3_1/user02/r02024/testSpace/quartus/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
Lab0
Use Existing Project Settings...
2
Lab0.bdf
Save
3
Save as type:
Block Diagram/Schematic Files (*.bdf)
Cancel
Add file to current project
Tasks
Flow:
All
<<Search>>
Type
ID
Message
Messages
System
Processing
1, 263
0%
00:00:00

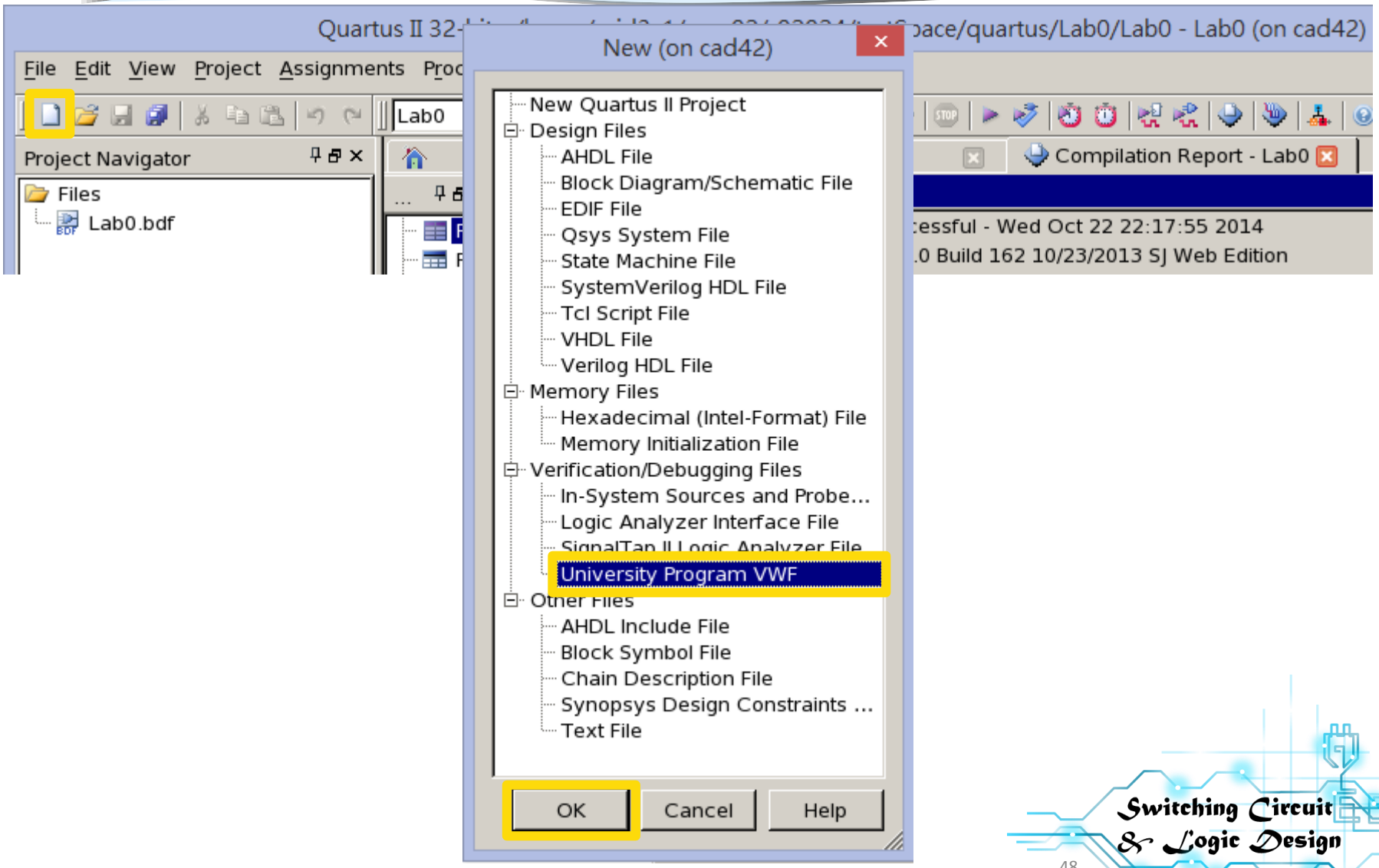
Save it


Quartus II 32-bit - /home/raid3_1/user02/r02024/testSpace/quartus/Lab0/Lab0 - Lab0 (on cad42)
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
Lab0
Project Navigator
Files
Lab0.bdf
Hierarchy
Files
Home
Lab0.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

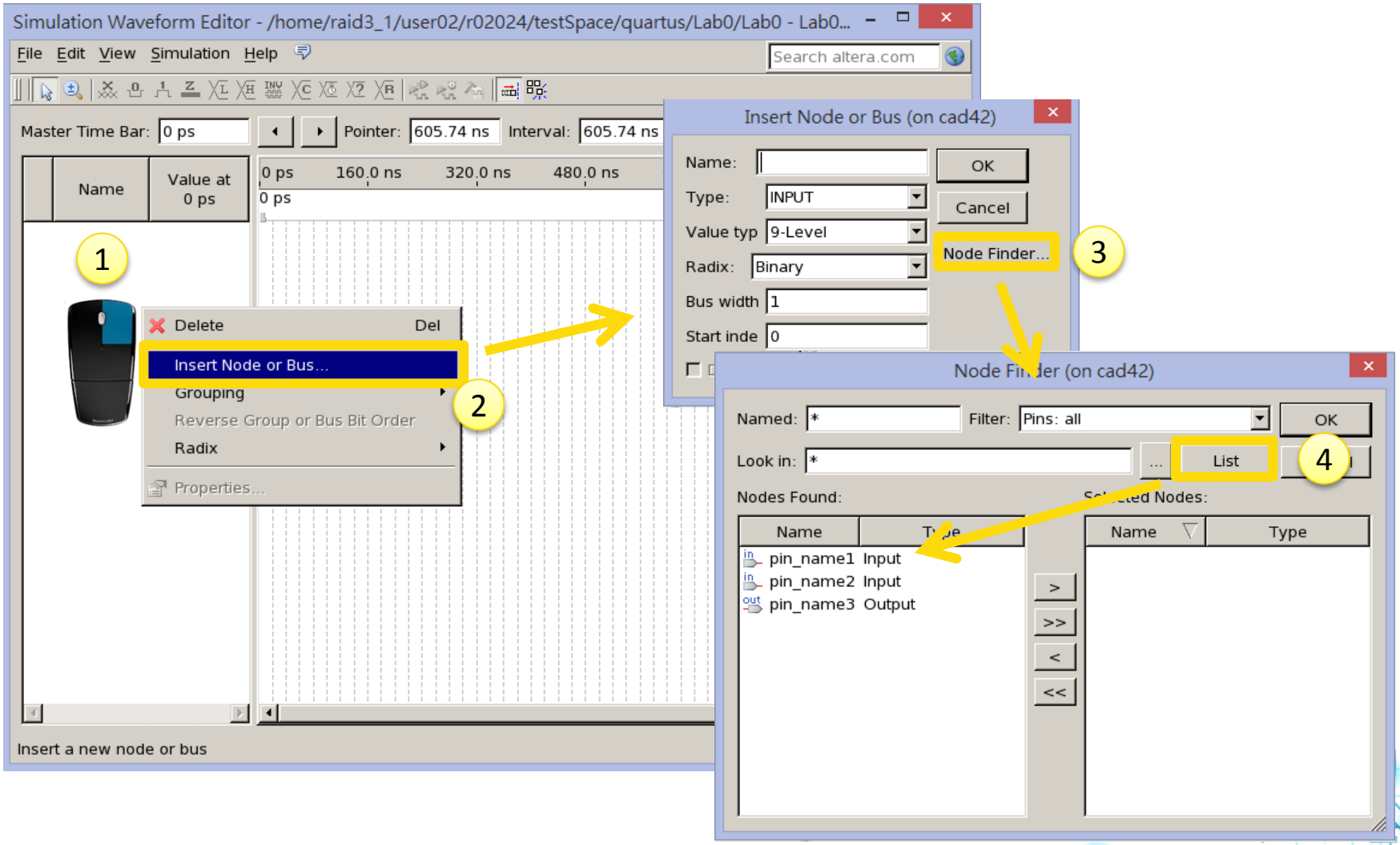
Compile


Quartus II 32-bit - /home/raid3_1/user02/r02024/testSpace/quartus/Lab0/Lab0 - Lab0 (on cad42)
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

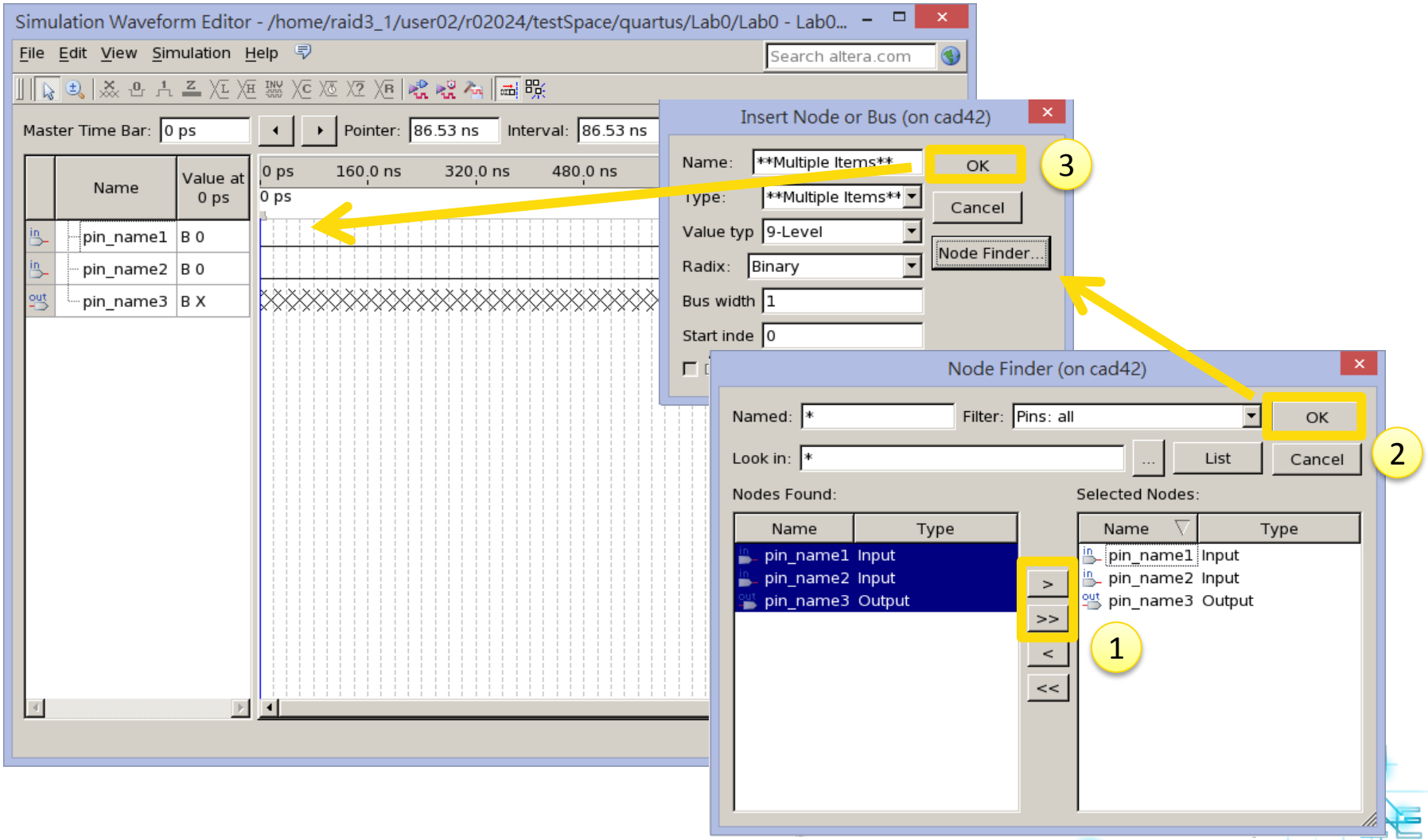
Set input signal



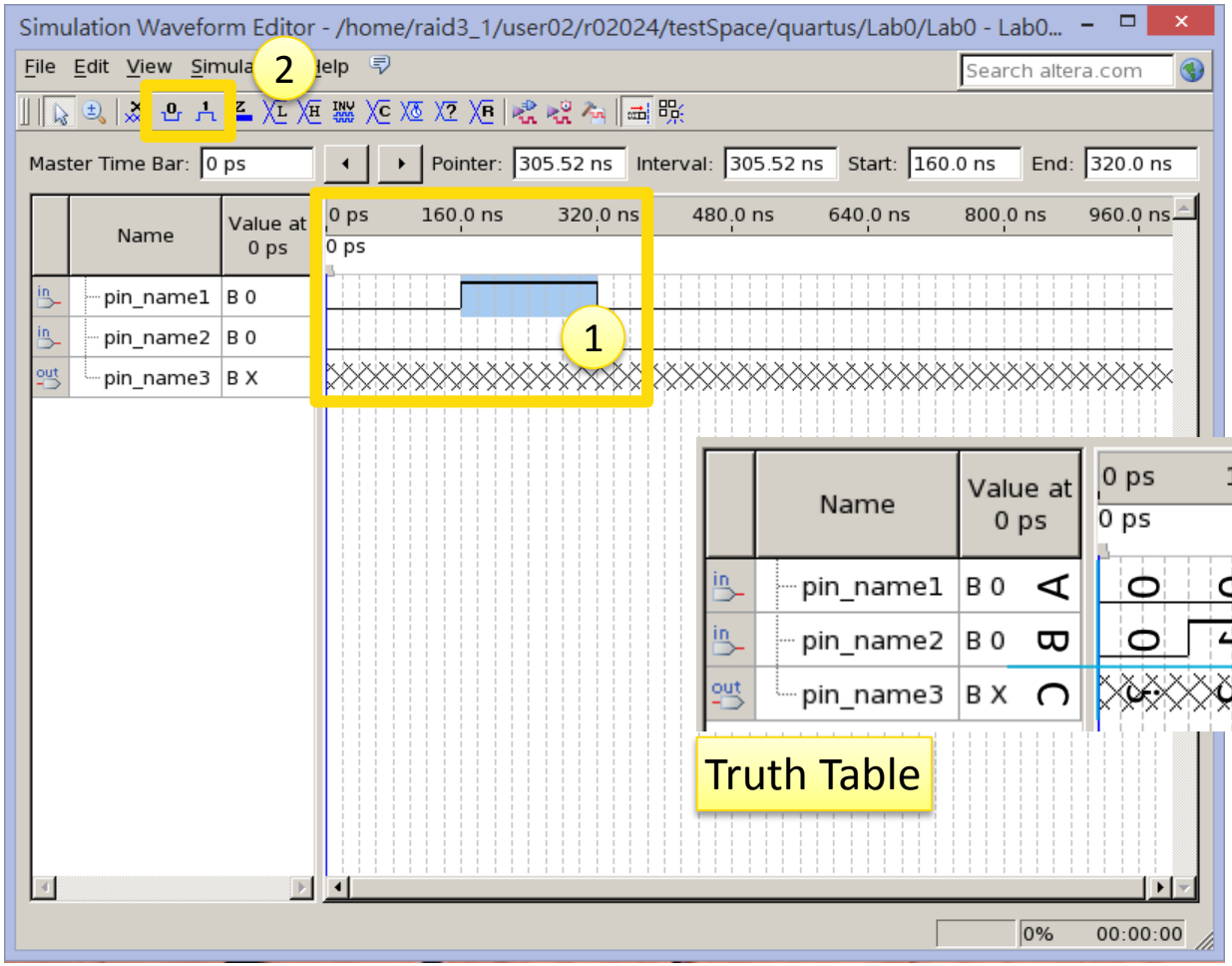
Set input signal



Set input signal

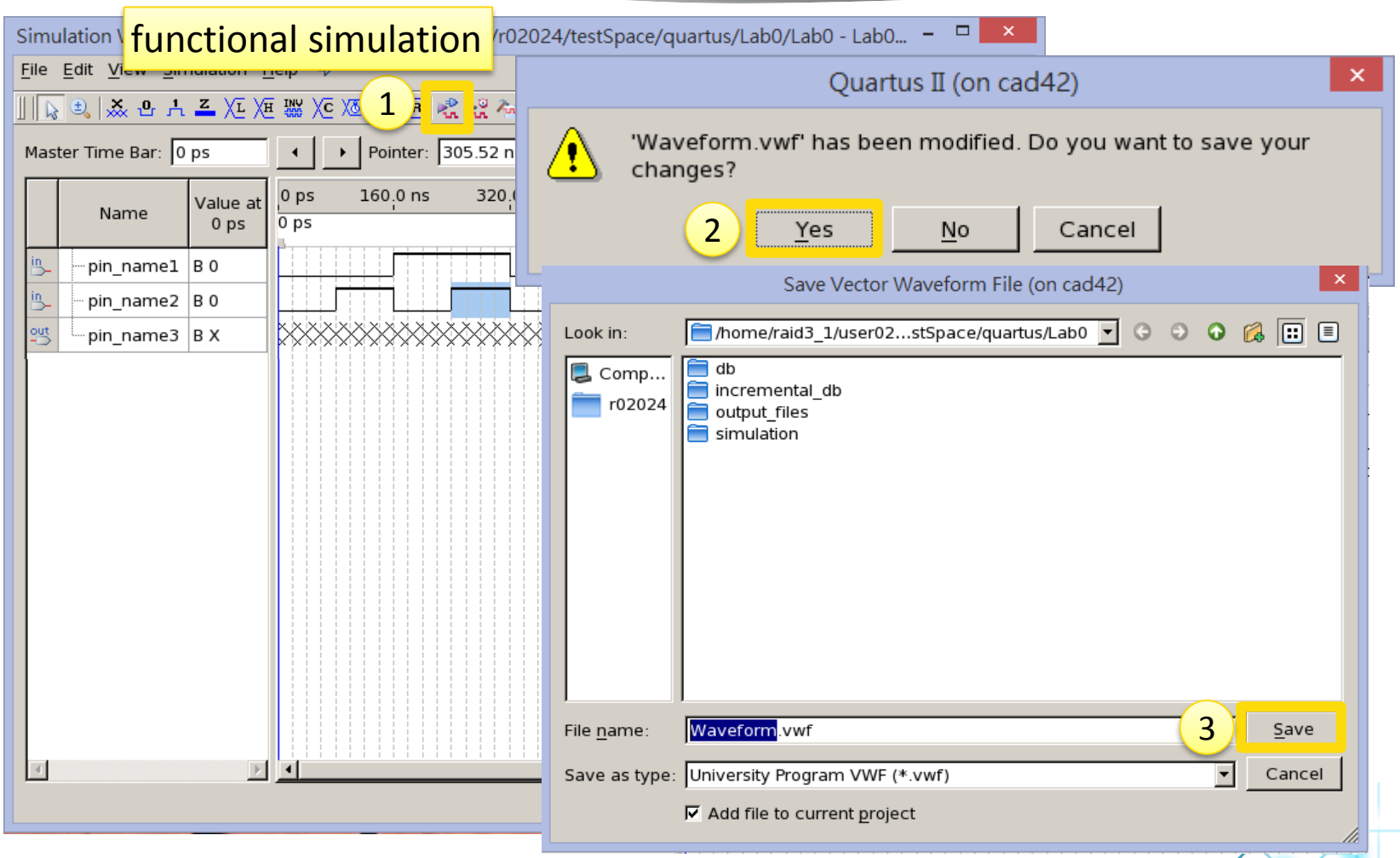


Set input signal

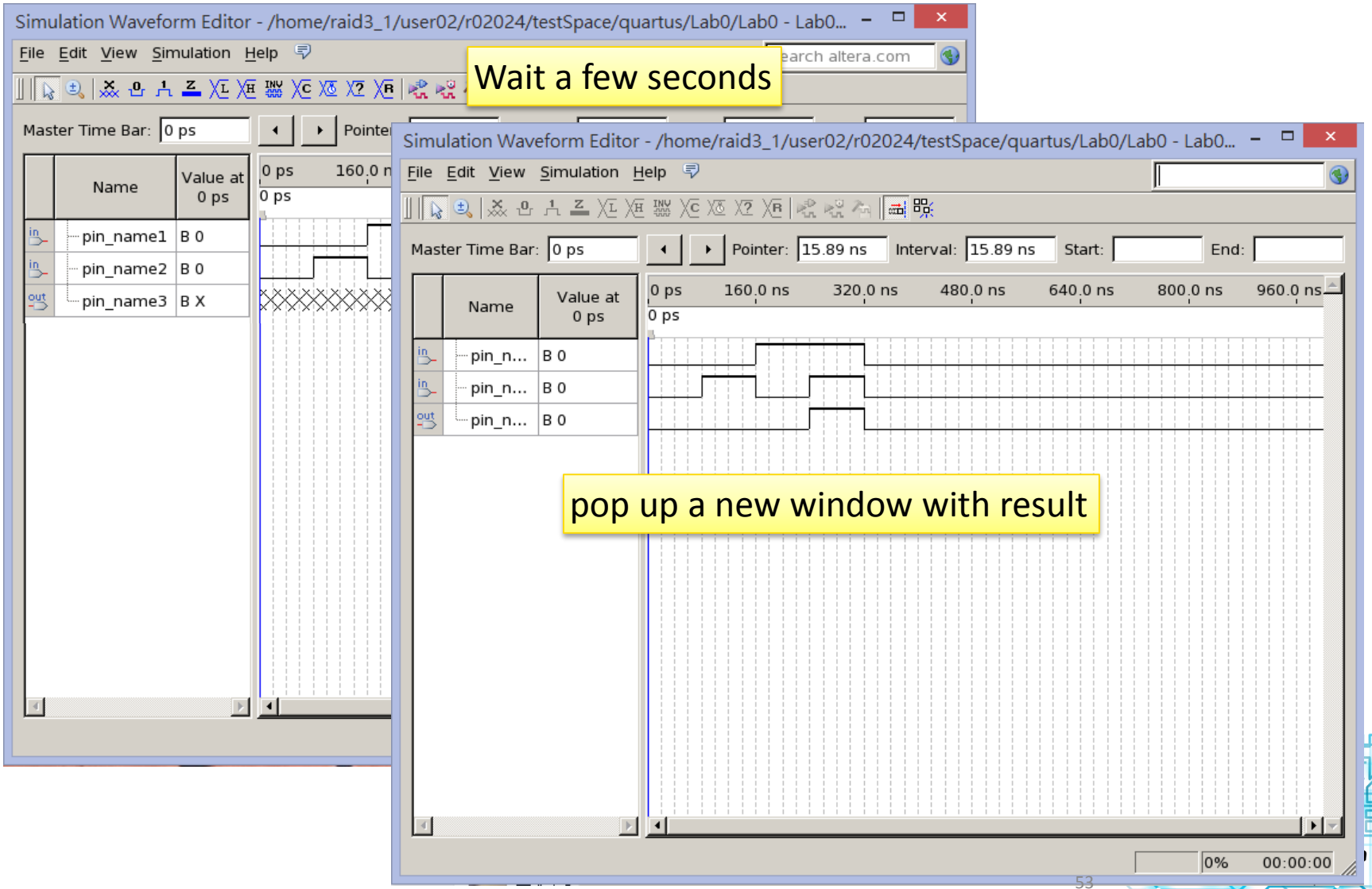


A	B	C
0	0	?
0	1	?
1	0	?
1	1	?

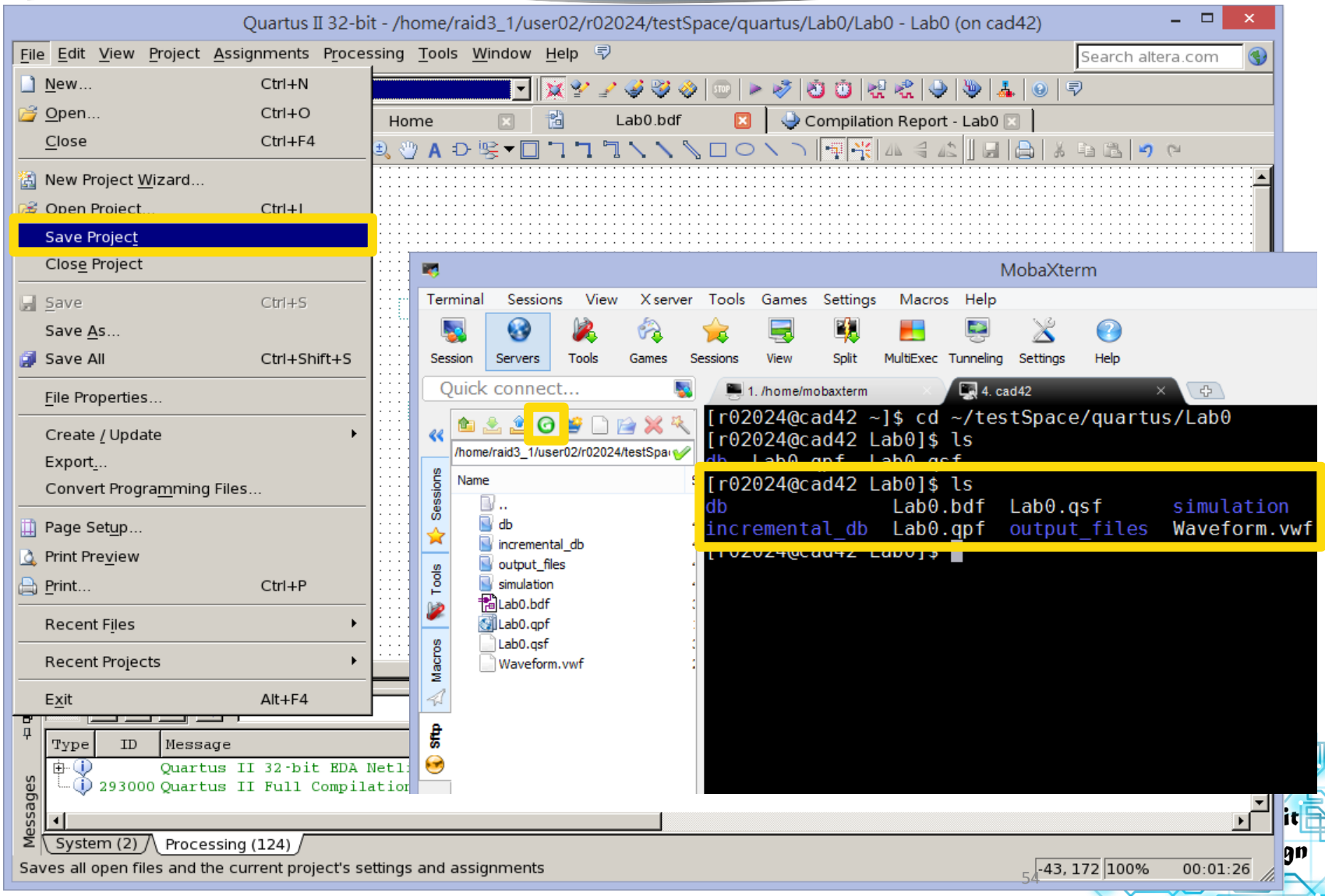
Simulation


functional simulation
Simulation W
/r02024/testSpace/quartus/Lab0/Lab0 - Lab0...
File Edit View Simulation Help
1
Master Time Bar: 0 ps
Pointer: 305.52 n
Name
Value at 0 ps
0 ps
160.0 ns
320.
pin_name1 B 0
pin_name2 B 0
pin_name3 B X
Quartus II (on cad42)
'Waveform.vwf' has been modified. Do you want to save your changes?
2
Yes
No
Cancel
Save Vector Waveform File (on cad42)
Look in: /home/raid3_1/user02...stSpace/quartus/Lab0
Comp...
r02024
db
incremental_db
output_files
simulation
File name: Waveform.vwf
3
Save
Save as type: University Program VWF (*.vwf)
Cancel
Add file to current project

Simulation


Simulation Waveform Editor - /home/raid3_1/user02/r02024/testSpace/quartus/Lab0/Lab0 - Lab0...
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
Wait a few seconds
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

Lab1: 4 bits Full Adder

- Gate-level schematic design flow:
  1. Divide system - Divide big design to system of small modules
     - For every small modules, do the following steps:
  2. Truth table - Think detailedly 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 it first - Be sure meet the requirements (ex. gate count)
  6. Quartus design - Implement it (.bdf)
     - You can pack partial circuit as sub-module by .bsf + .bdf
  7. Verify & debug - Test some typical patterns, simulate it by .vwf

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 to design

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

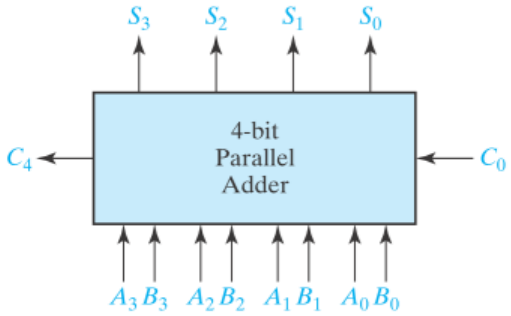
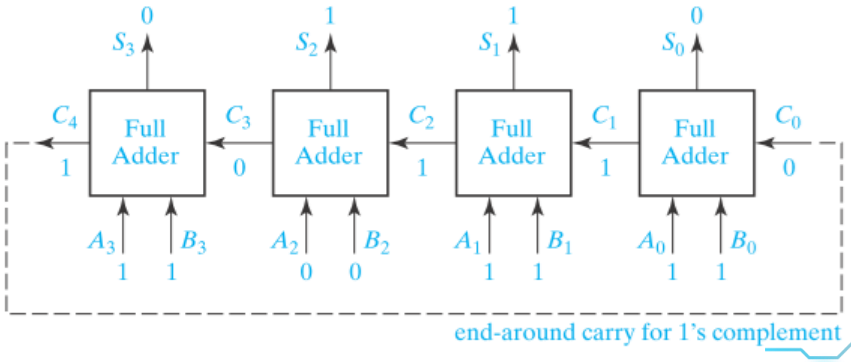


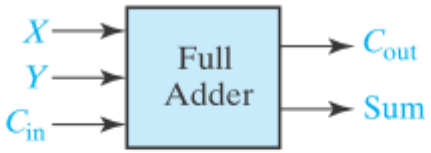
FIGURE 4-3
Parallel Adder Composed of Four Full Adders



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 detailedly 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

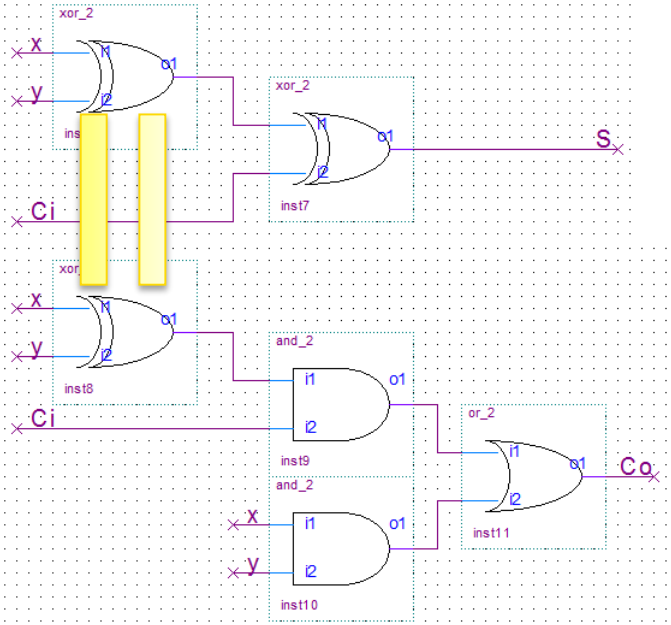
$S = x \oplus y \oplus Ci$

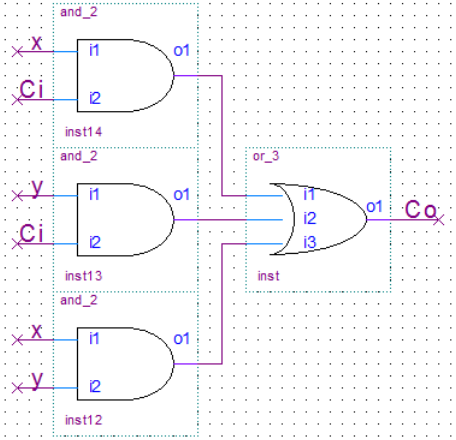
Step 4: Boolean Expression

- 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

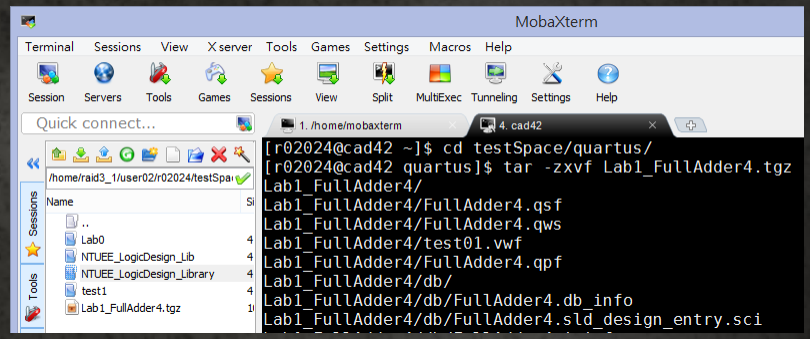




GIVE IT A TRY !

Step 6: Quartus Design

- Let's start!!
  - ☑ Upload Lab1_FullAdder4.tgz to your workstation



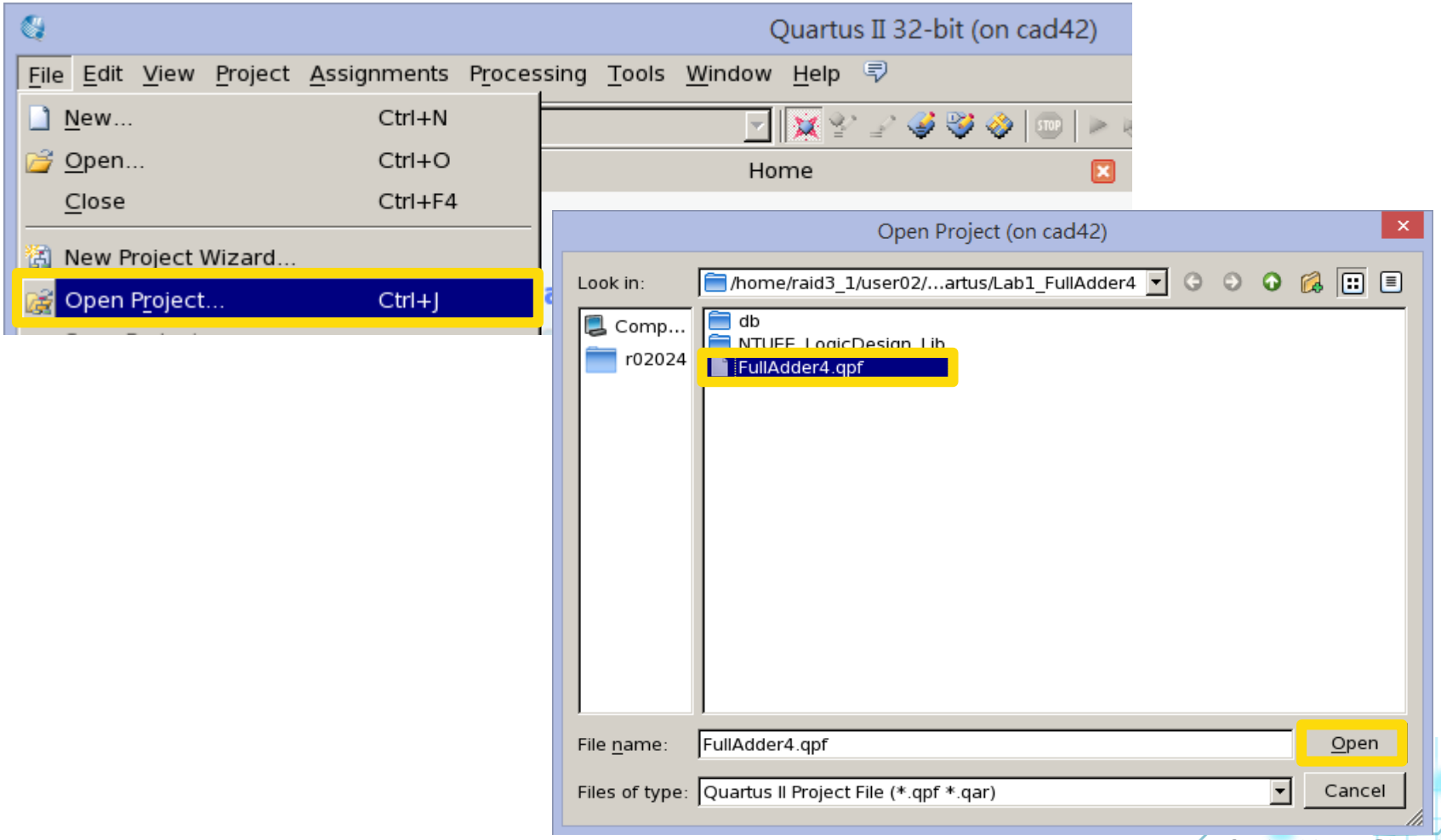
  - ☑ Decompress it

```
> tar -zxvf filename.tgz
```

  - ☑ Start Quartus II

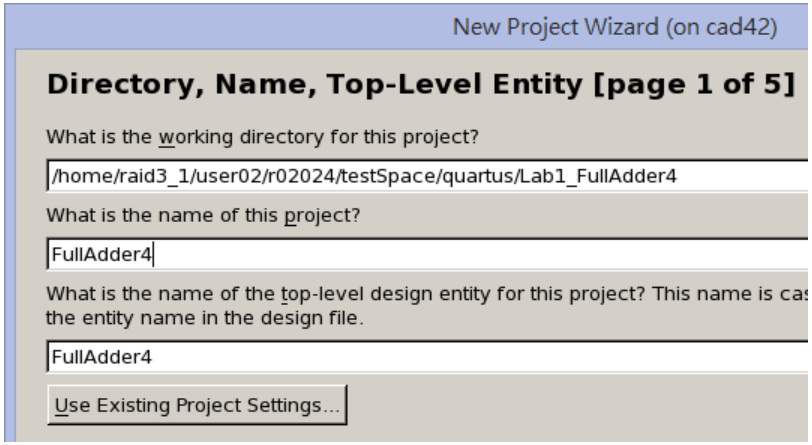
```
tcsh
source /home/raid5_4/raid2_3/solaris/Quartus/.cshrc
quartus &
```

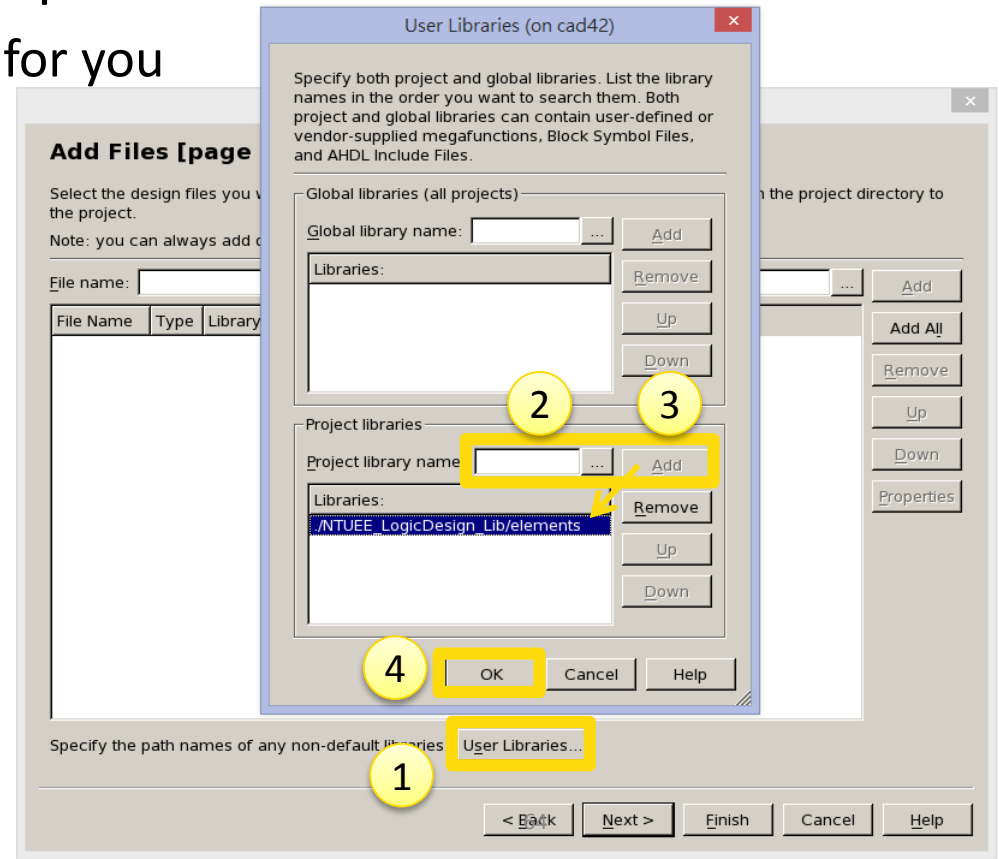
Open Project



Important things about this project

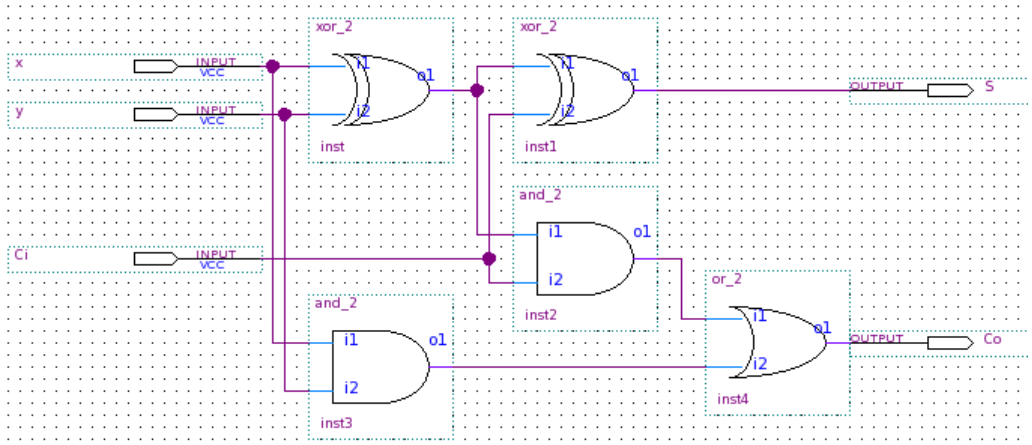
- This project is built with top design named “FullAdder4”
- In all the Labs, we restrict you:
  only able to use logic gates TA provided
  - TA already put them as library for you
  - If you want to set it yourself in the future, do this → when building the project

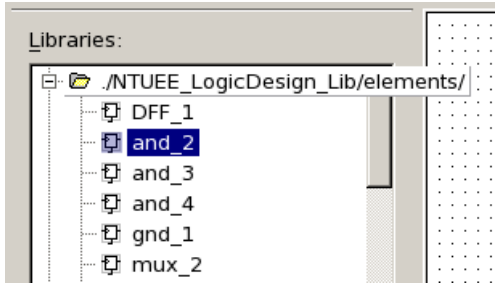




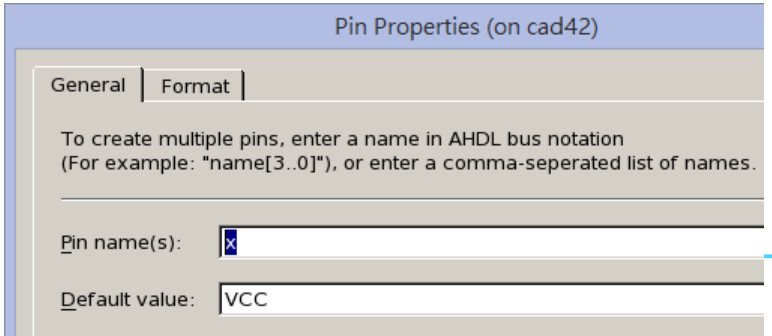
Build the FullAdder1.bdf

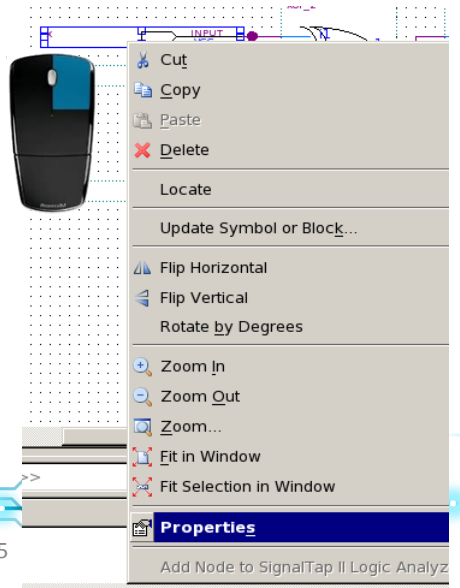
- File→New…→Block Diagram/Schematic File→OK
  - Implement it only use gates under NTUEE_LogicDesign_Lib





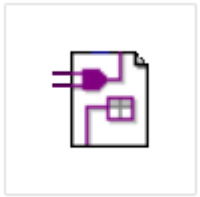
- Important hint!!
  - right click on the I/O pin→Properties to name it
  - The name of pin is important!!





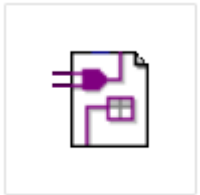
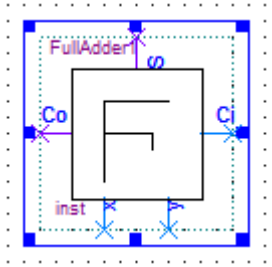
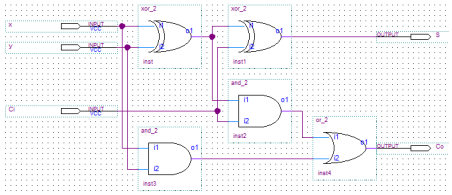
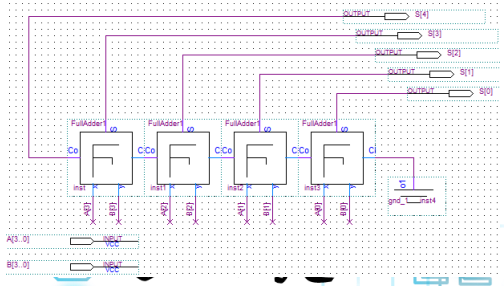
Make FullAdder1 a “Module”

- 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. FullAdder4 to use it
C++ Object	Object Class (header file) with data, interface  MyClass.h	Object Class (source file)  MyClass.cpp	Use class included by header in another cpp ex. main.cpp

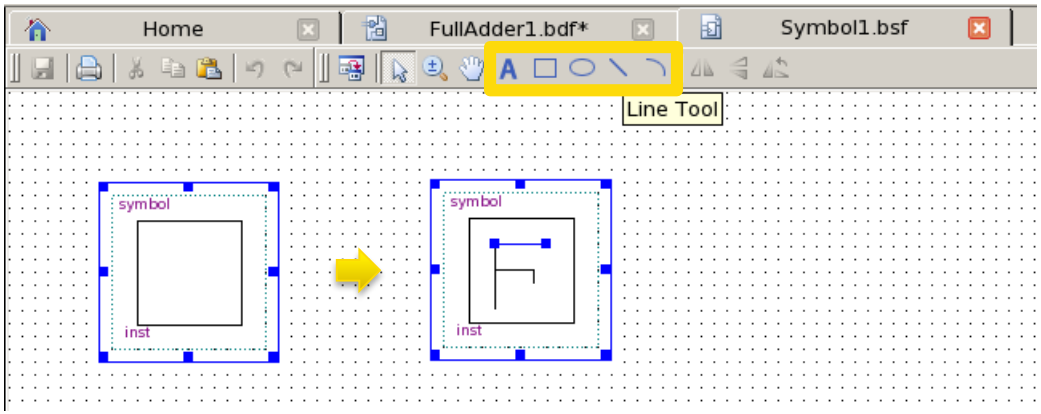
Make FullAdder1 a “Module”

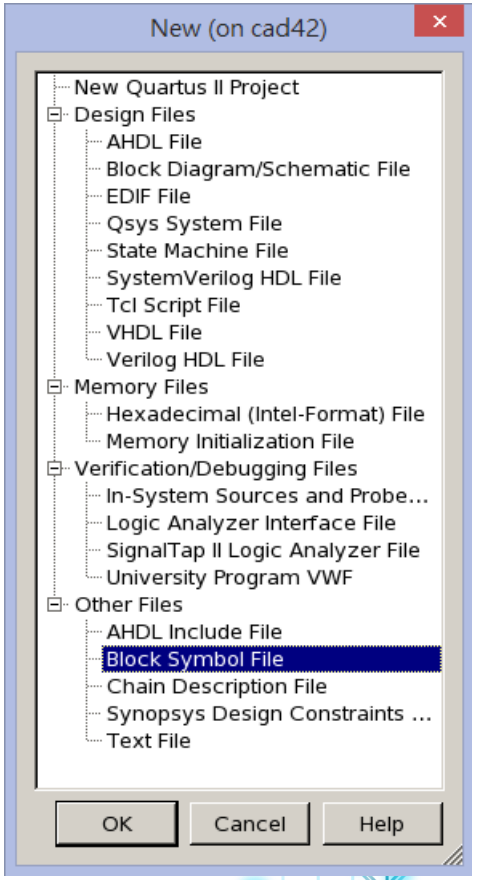
- We want to use the “FullAdder1” 4 times, we can declare it as a module (like C++ class) and generate many instance!!
- All you need is a .bsf file

	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. FullAdder4 to use it
			

Draw a FullAdder1.bsf (1/3)

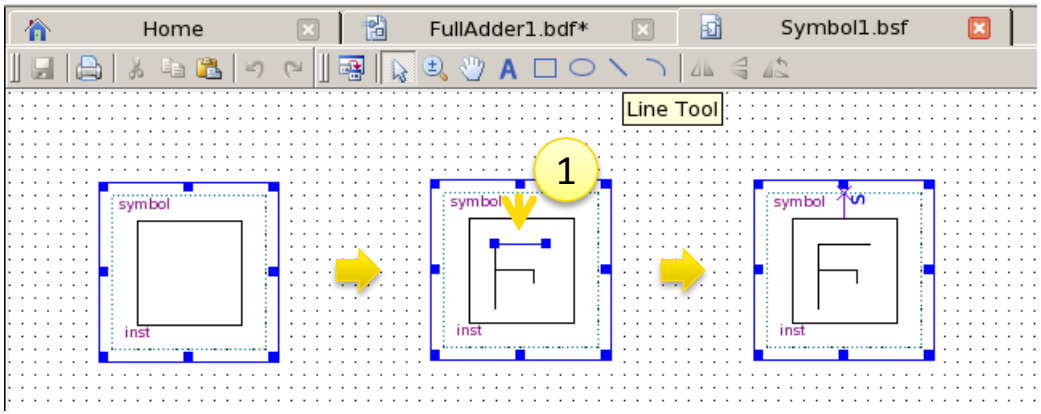
- File→ New…→Block Symbol File→OK
- Draw it as you wish by drawing tools

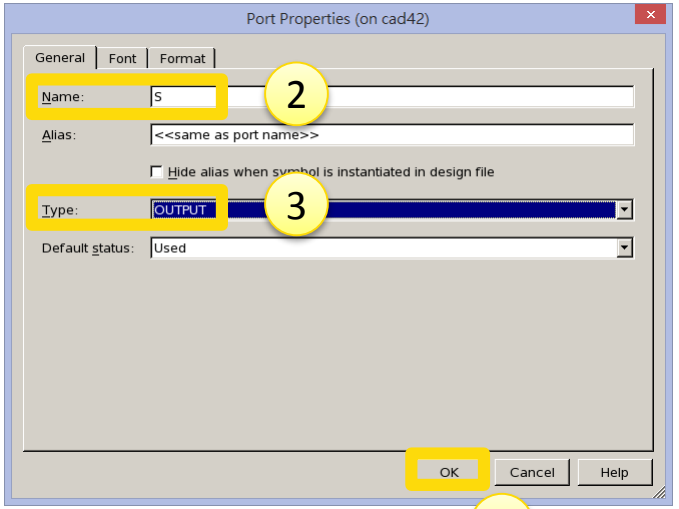




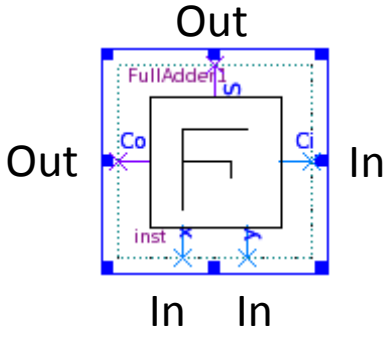
Draw a FullAdder1.bsf (2/3)

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



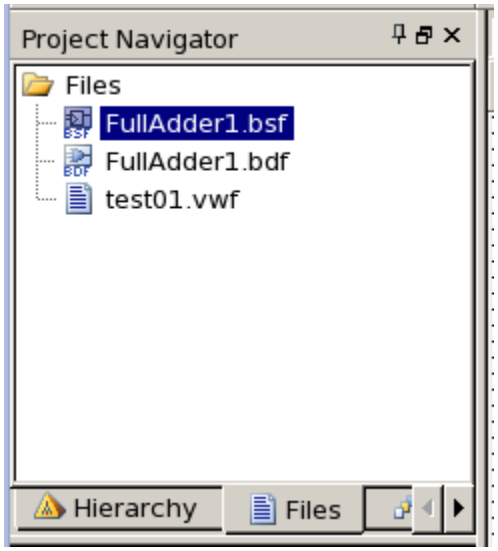


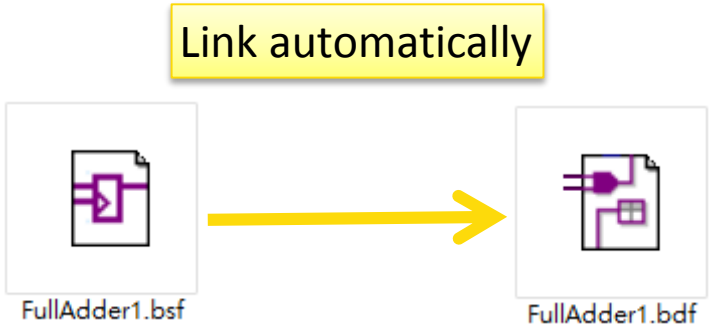
- Repeat to add all pins properly



Draw a FullAdder1.bsf (3/3)

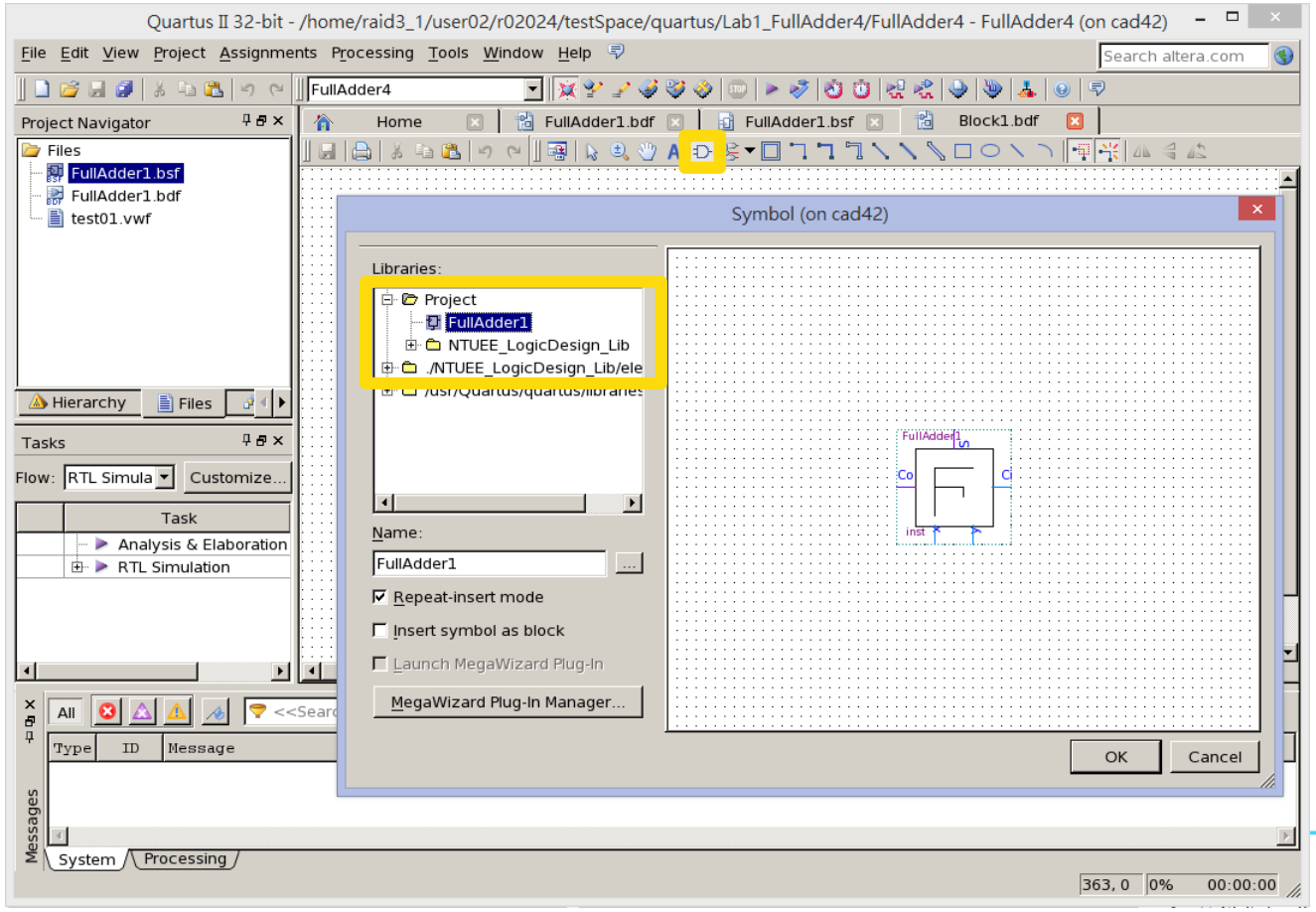
- Save this .bsf as FullAdder1.bsf
  - Important!! Names are mattered for Quartus
  - The .bsf will search for implementation file with the same name





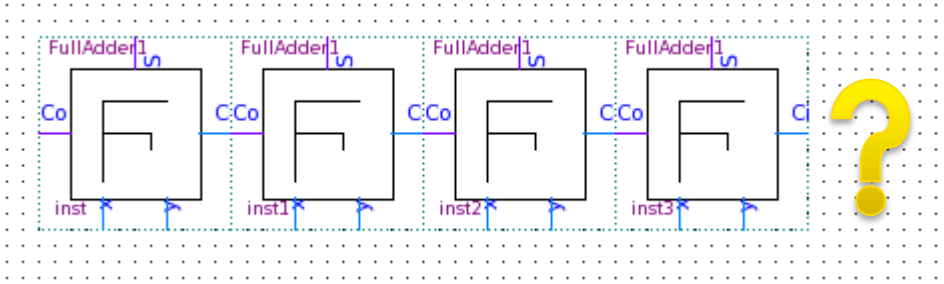
The Top Design: FullAdder4.bdf

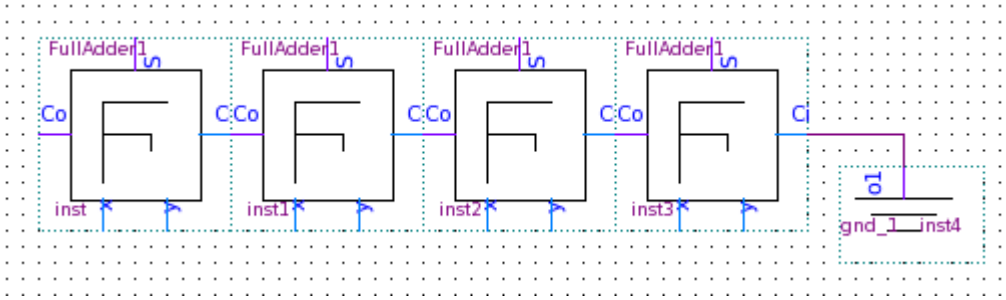
- File→New…→Block Diagram/Schematic File→OK
- From now on, you can use the “FullAdder1” module



0 and 1, Ground and VCC

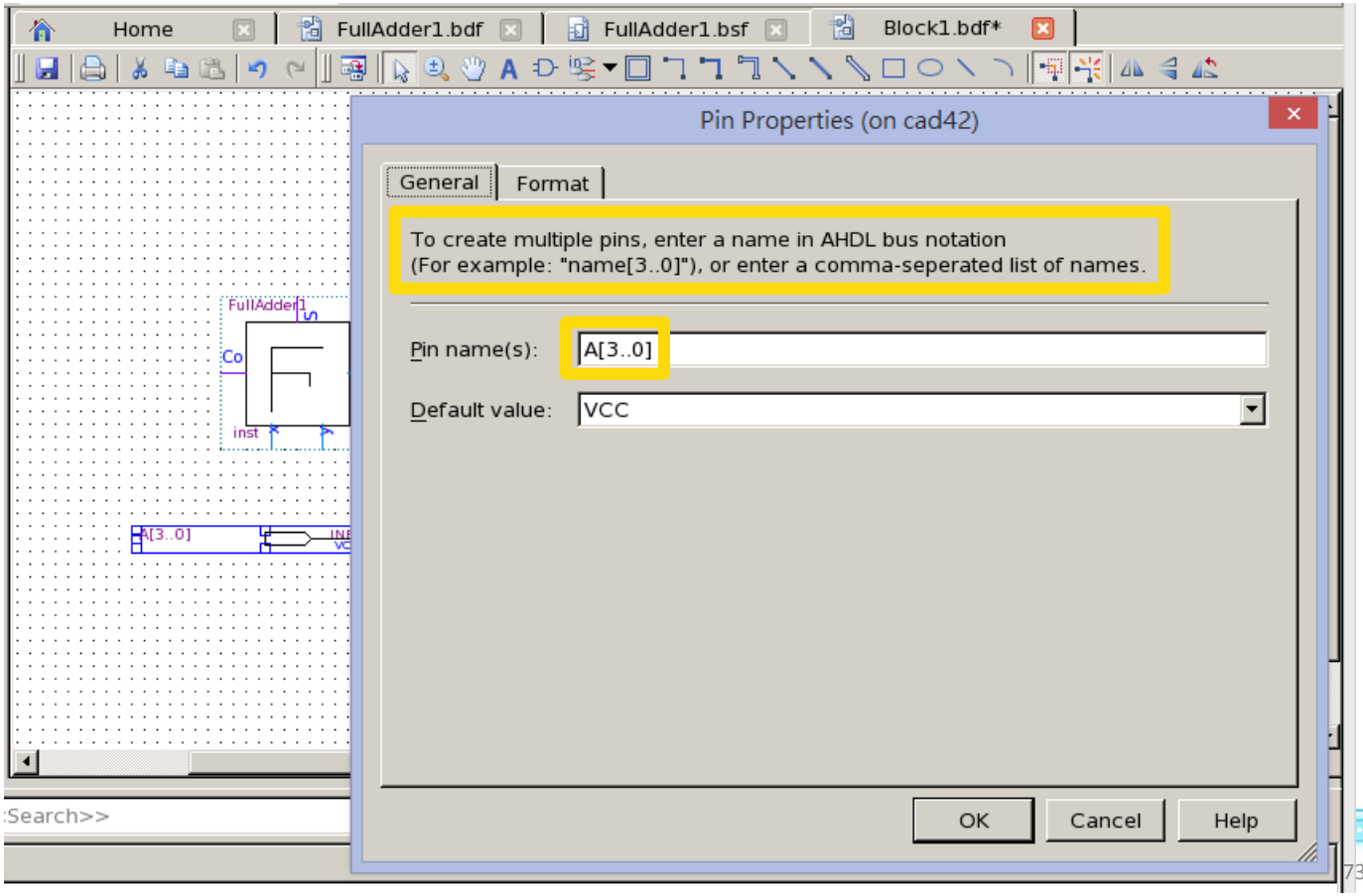
- You can
  - use ground (gnd_1) as 0
  - use voltage circuit (vcc_1) as 1





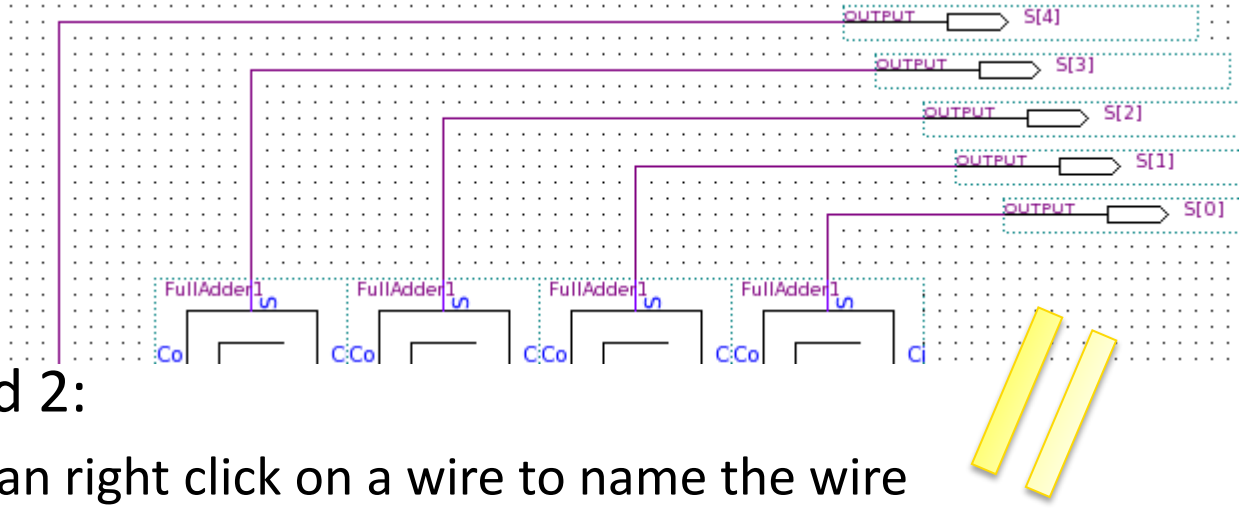
Array of pins / wires

- 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]

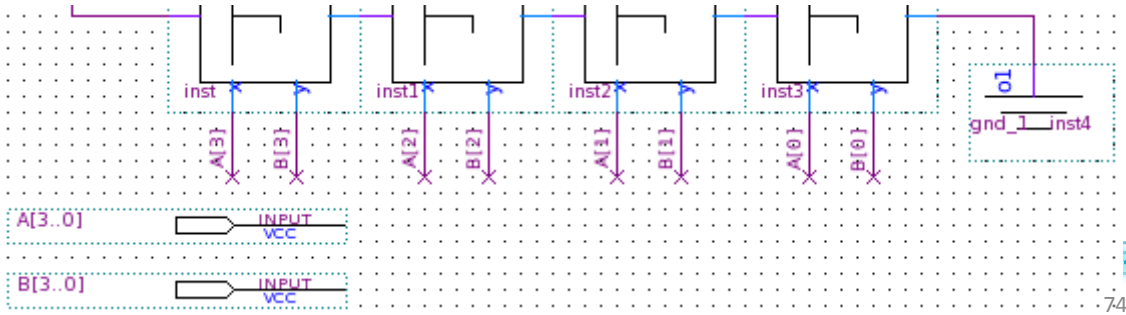


Array of pins / wires

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



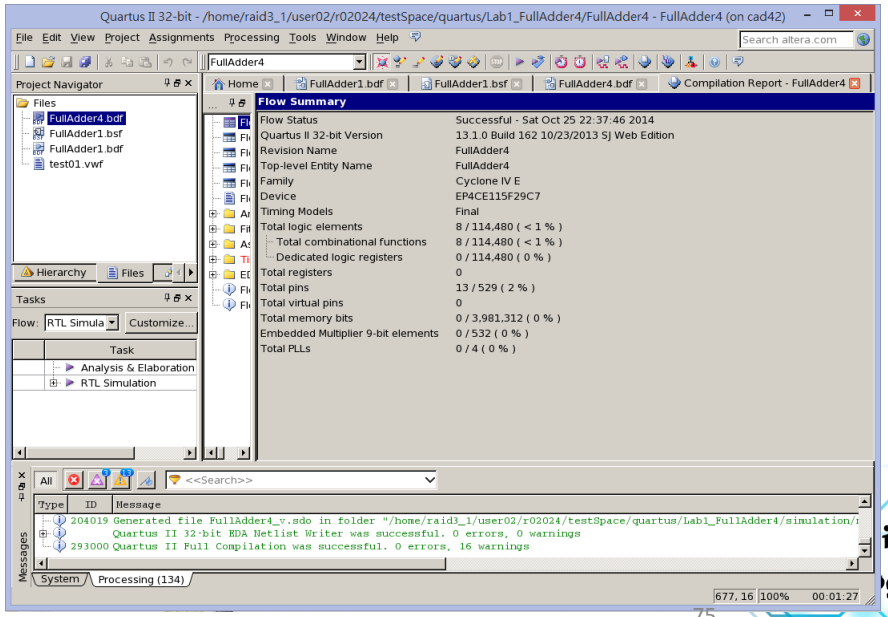
- Method 2:
  - You can right click on a wire to name the wire
  - Wires / pins with the same name "are linked implicitly"



The Top Design: FullAdder4.bdf

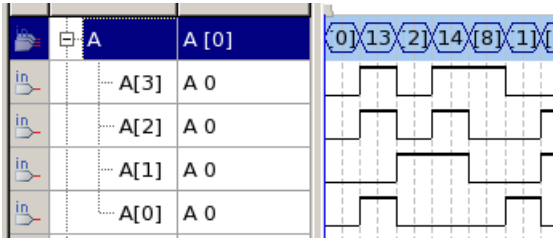
- Remember to save this bdf as FullAdder4.bdf
  - The “name” that our top design is set when project created
- File→Save poject

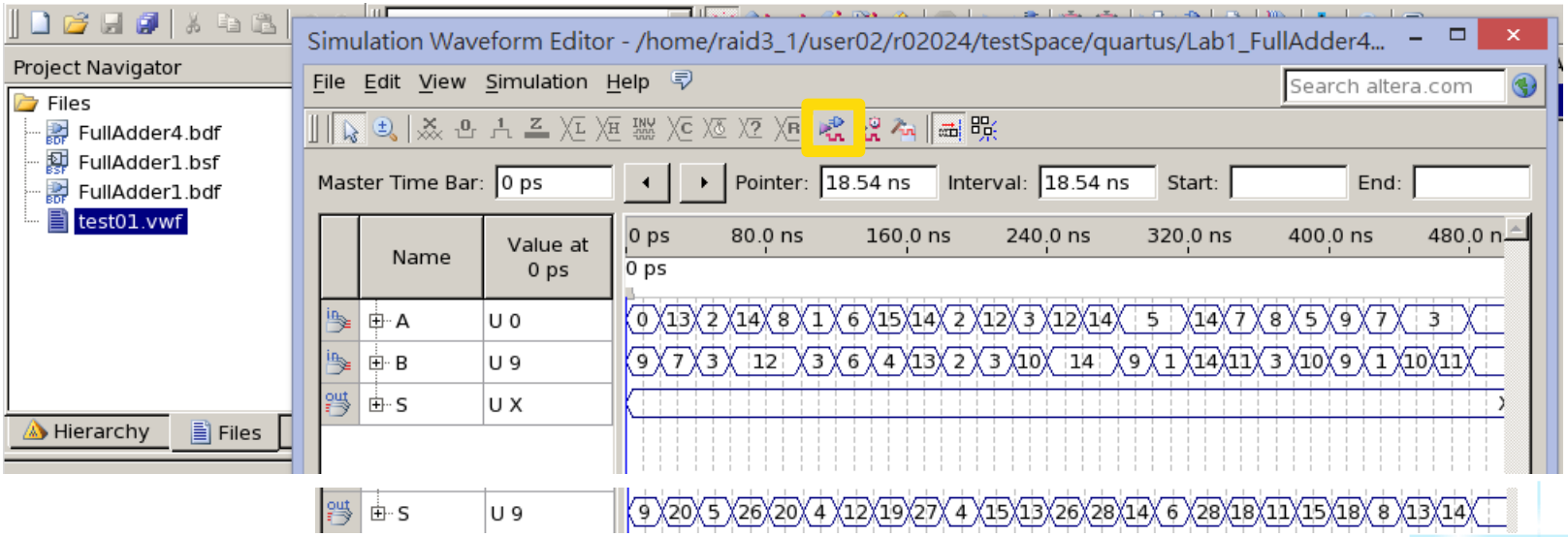
- Start Compilation
  - Check no error exists
- File→Save poject



Step 7: Verify & Debug

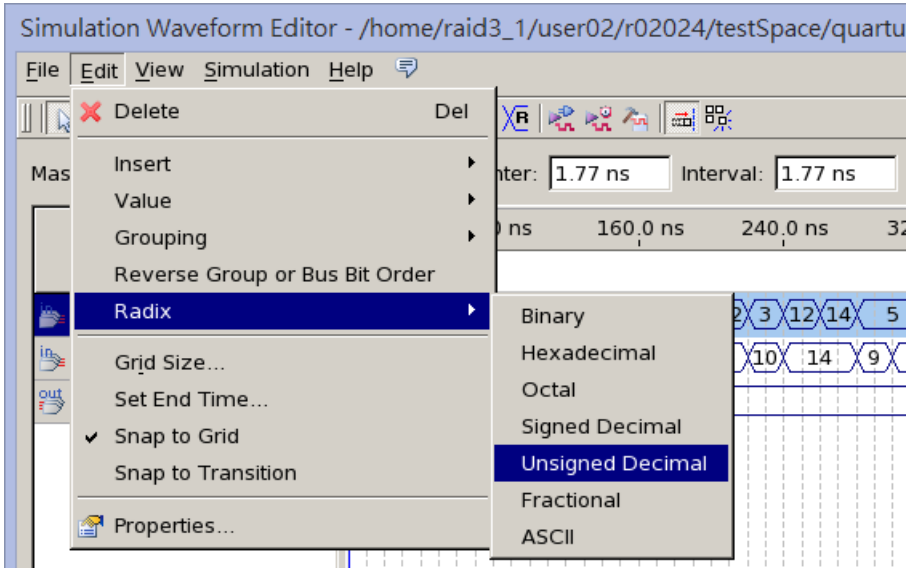
- Test some typical patterns, simulate it by .vwf
- We are “requested” (given) a test01.vwf file
  - We must follow the I/O pins name of it
  - And pass all the patterns

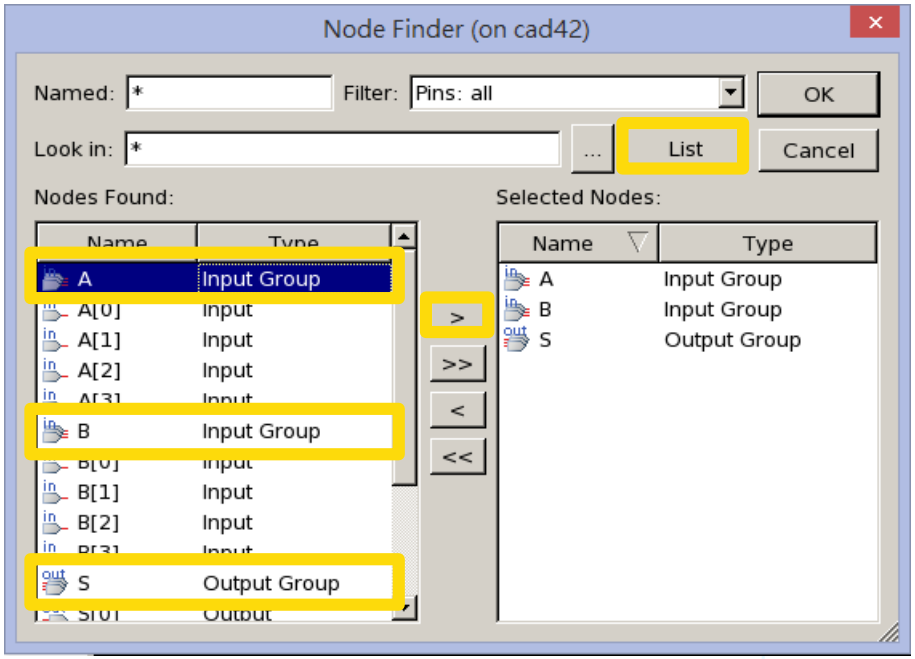




vwf Advanced

- How can we make a vwf like this?
  - You can pick “wire group” when using node finder
    - Right→Insert Node or Bus…
      →Node Finder… →List
  - You can set radix to change base

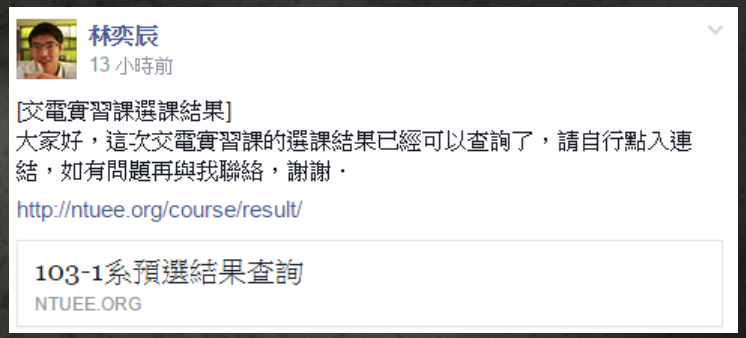




GIVE IT A TRY !

Homework: FullSubtracter4

- Do it at 交電實習時段!!
  - ☑ TA will give you the target file: Lab1HW_FullSubtracter.tgz
  - ☑ Decompress it
  - ☑ Start Quartus II, do the design steps:
    - Divide system
    - Truth table, Simplify (K-map maybe…), Boolean expression
    - Draw it first and check the constrains
    - Quartus design
    - Verify & debug
  - ☑ Top design: FullSubtracter4
  - ☑ Pass the test01.vwf → You need to show the results to TA



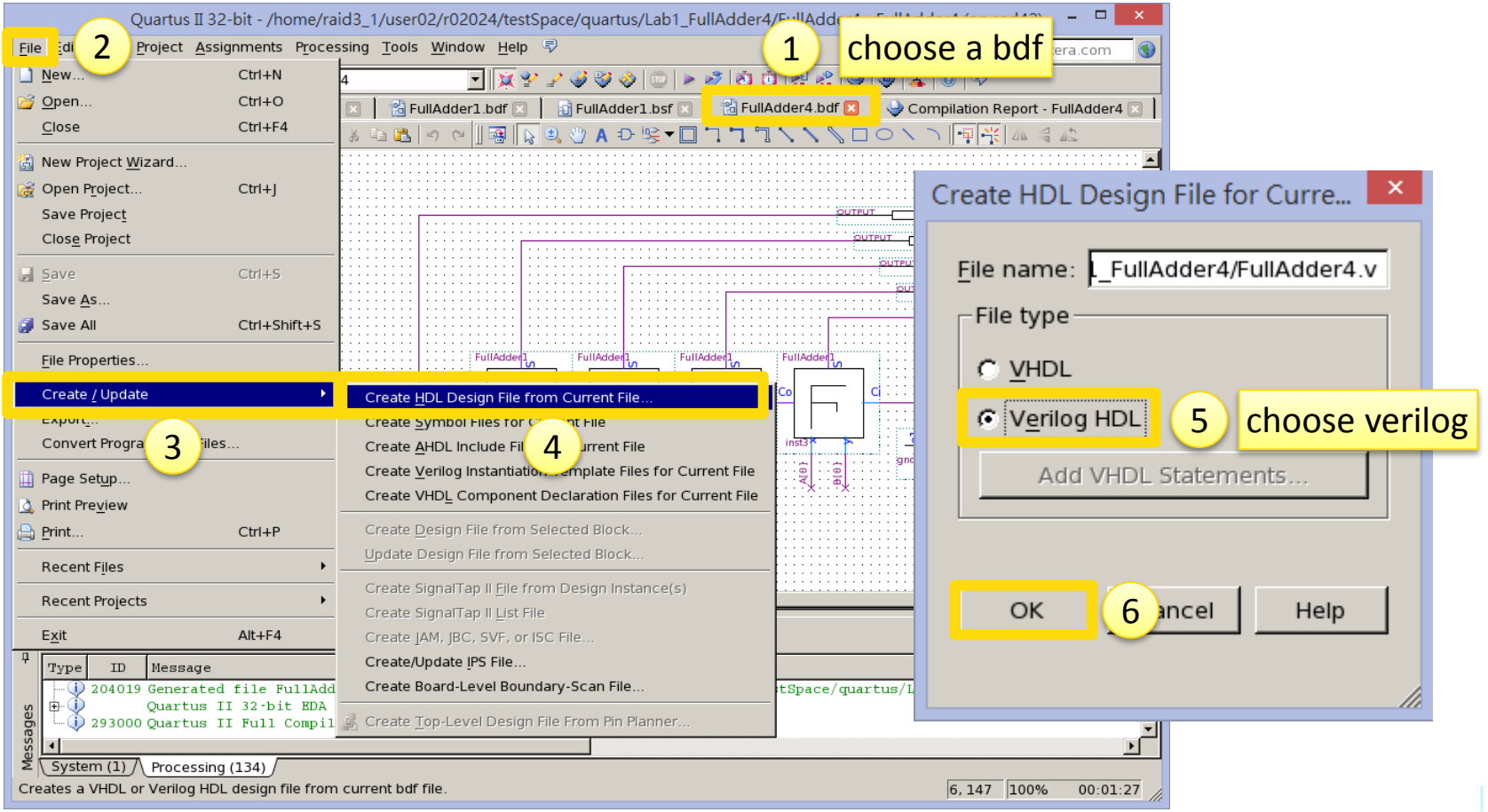
Quick Preview

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"

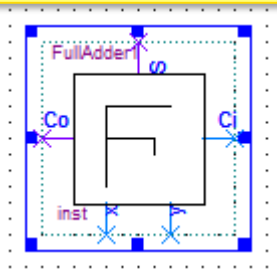
Generate Verilog Design File from Schematic

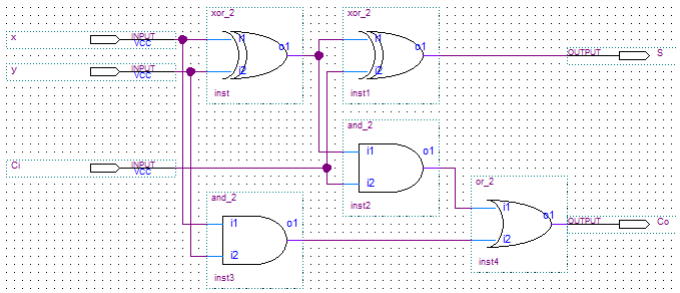


An Example - FullAdder

- Interconnections of logic elements

Module interface declaration





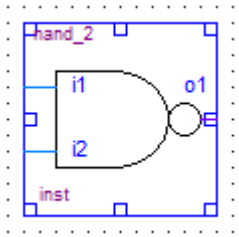
```
module FullAdder1(Ci,x,y,S,Co);
input wire  Ci;
input wire  x;
input wire  y;
output wire S;
output wire Co;

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

An Example - NAND_2

- Descripting the behavior of logic elements

Module interface declaration

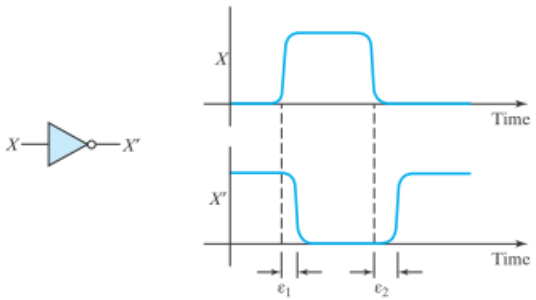



```
module nand_2(
input i1,
input i2,
output o1
);
    assign o1 = ~(i1&i2);
    specify
        (i1 => o1) = (3:3:4.5,3:3:5);
        (i2 => o1) = (3:3:4.5,3:3:5);
    endspecify
endmodule
```

Descripting the functional logic in higher level language (like C++)

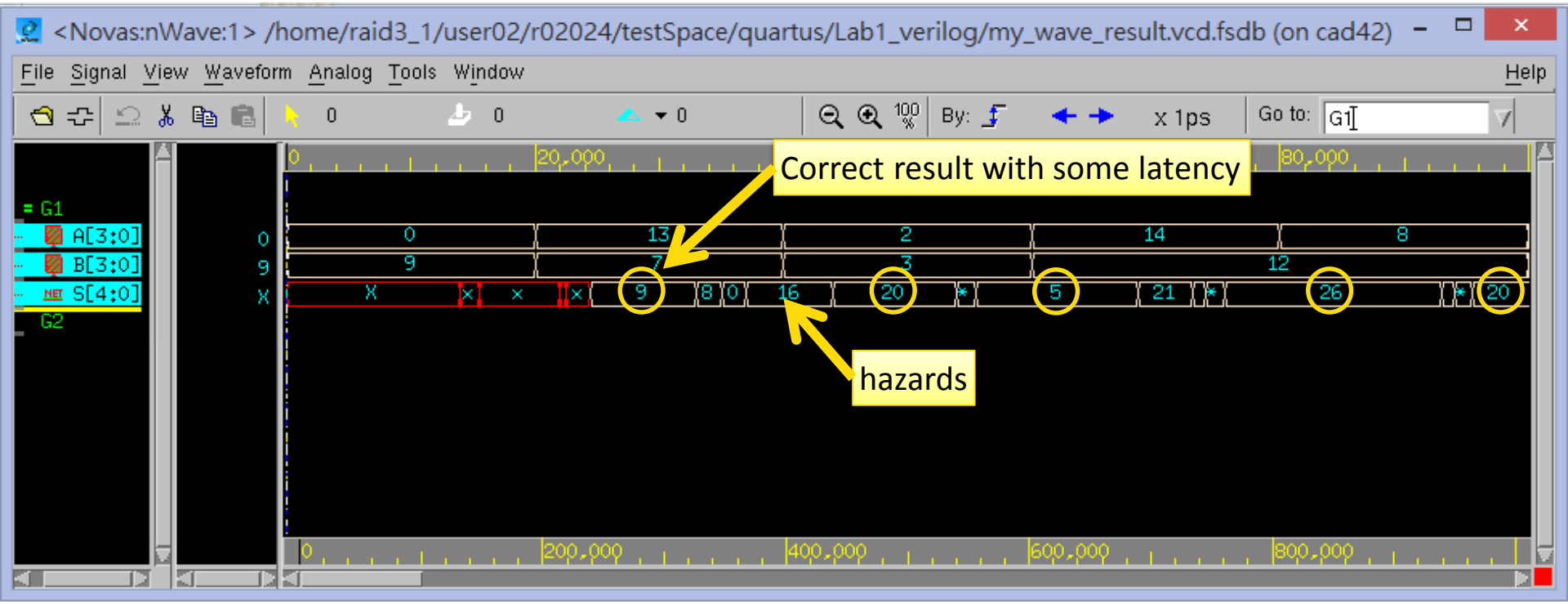
Descripting the timing / delay information for simulation

FIGURE 8-4
Propagation Delay in an Inverter



(min T_{PLH} : avg T_{PLH} : max T_{PLH} , min T_{PHL} : avg T_{PHL} : max T_{PHL})
T_{PLH}: propagation delay from low to high
T_{PHL}: propagation delay from high to low

nWave



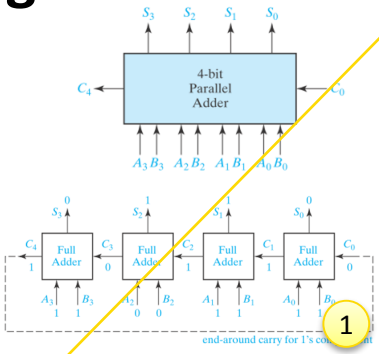
Pretty like simulation result as .vwf does
We can define delay information in verilog files
So there are hazards & latency (Not functional simulation only)

Summary

Schematic Combinational Circuit Design

General Design Flow

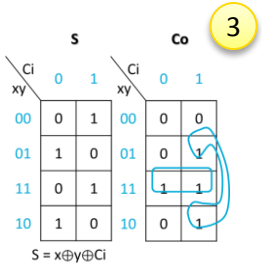
1. Divide system
2. Truth table
3. K-map
4. Boolean expression
5. Draw it first
6. Design with tools
7. Verify & debug



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

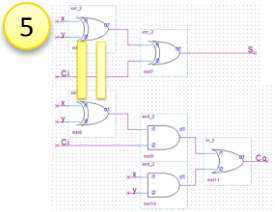
2



3

4

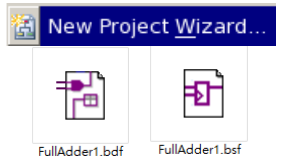
– S = (x⊕y)⊕Ci
– Co = (x⊕y)·Ci + x·y

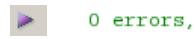


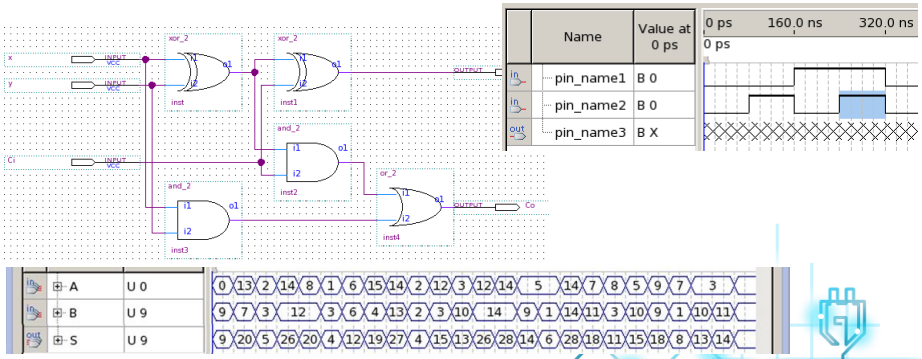
5

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

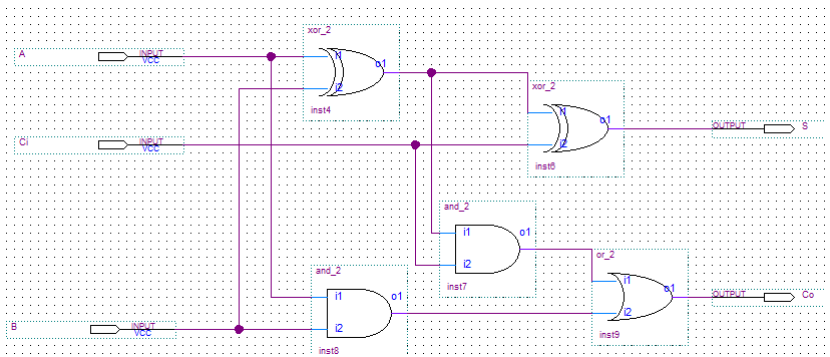






Schematic v.s. HDL

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. RT level)

Better ask twice than lose you way once.

Q&A

Reference

Acknowledgement

- Authors of Basic Logic Design via Verilog HDL
  - Ver. 1: Chen-han Tsai
  - Ver. 2: Chih-hao Chao
  - Ver. 3: Xin-Yu Shi
  - Ver. 4: Bo-Yuan Peng
  - Ver. 5: Chieh-Chuan Chiu & Chieh-Chi Kao
  - Ver. 6: Yu-Hao Chen & Ming-Chun Hsiao
  - Ver. 7: Yu-Hao Chen & Cheng-Rung Tsai

Reference

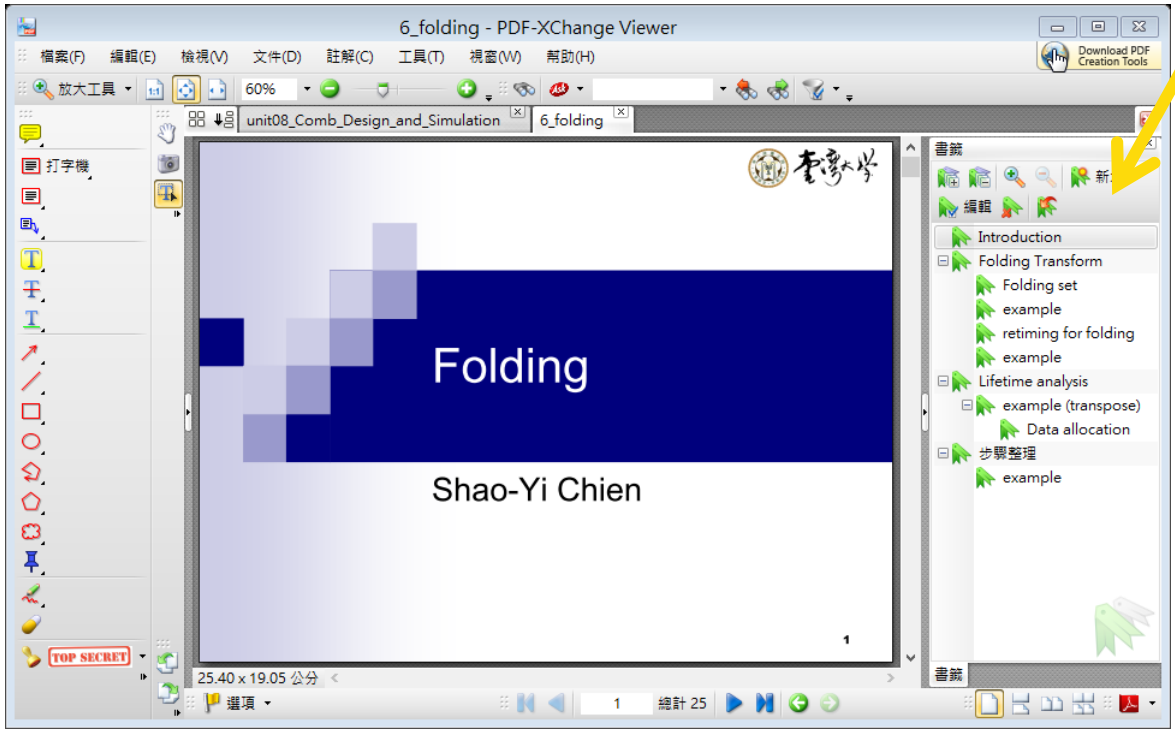
- Textbook
  - Fundamentals of Logic Design, Charles H. Roth, Jr., Larry L. Kinney
- Dclab lecture:
  - Verilog Coding Guideline, 吳柏辰
  - My First FPGA for Altera DE2-115 Board, 吳柏辰
- CVSD lecture:
  - Computer-aided VLSI System Design Linux / Unix Tutorial, 陳滿蓉

Useful Tools

PDF-XChange Viewer

- A charge-free software to “edit” pdf files
  - Useful for taking notes

You can add “book marks” to jump quickly



Notepad++ & Plug-ins

- Useful plug-ins:
  - Compare, HEX-Editor, NppFTP