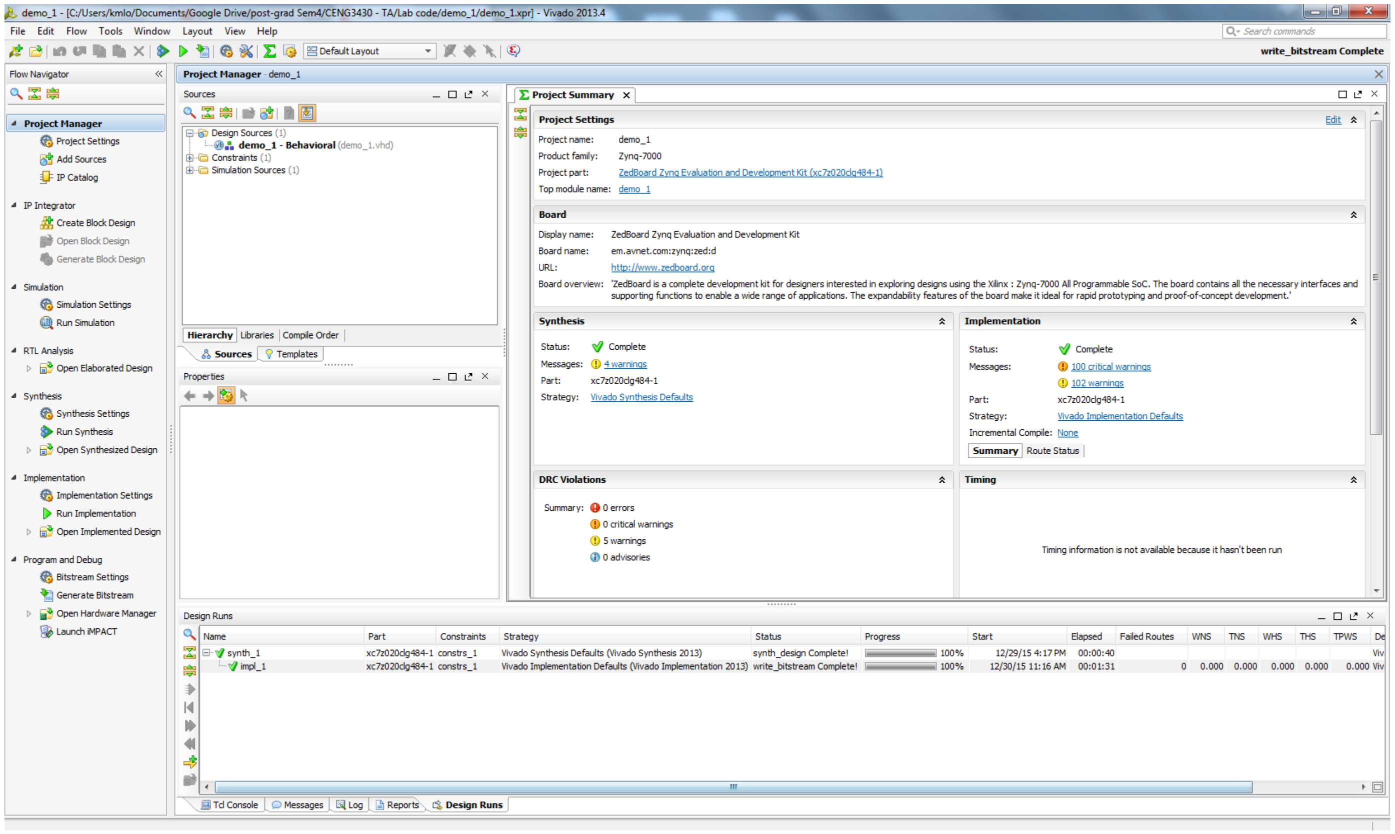
香港中文大學
The Chinese University of Hong Kong

CENG3430 Rapid Prototyping of Digital Systems

Lab01: Introduction to Vivado & Software Simulation

Vivado

- **Programming** using IDE for VHDL (Lab01)
- Running **simulation** to test your design (Lab01)
- **Downloading** the compiled code to FPGA (Lab02)



demo_1 - [C:/Users/kmlo/Documents/Google Drive/post-grad Sem4/CENG3430 - TA/Lab code/demo_1/demo_1.xpr] - Vivado 2013.4

write_bitstream Complete

Project Settings

Project name: demo_1
Product family: Zynq-7000
Project part: ZedBoard Zynq Evaluation and Development Kit (xc7z020clg484-1)
Top module name: demo_1

Board

Display name: ZedBoard Zynq Evaluation and Development Kit
Board name: em.avnet.com:zynq:zed:d
URL: http://www.zedboard.org
Board overview: 'ZedBoard is a complete development kit for designers interested in exploring designs using the Xilinx : Zynq-7000 All Programmable SoC. The board contains all the necessary interfaces and supporting functions to enable a wide range of applications. The expandability features of the board make it ideal for rapid prototyping and proof-of-concept development.'

Synthesis

Status: Complete
Messages: 4 warnings
Part: xc7z020clg484-1
Strategy: Vivado Synthesis Defaults

Implementation

Status: Complete
Messages: 100 critical warnings, 102 warnings
Part: xc7z020clg484-1
Strategy: Vivado Implementation Defaults
Incremental Compile: None

DRC Violations

Summary: 0 errors, 0 critical warnings, 5 warnings, 0 advisories

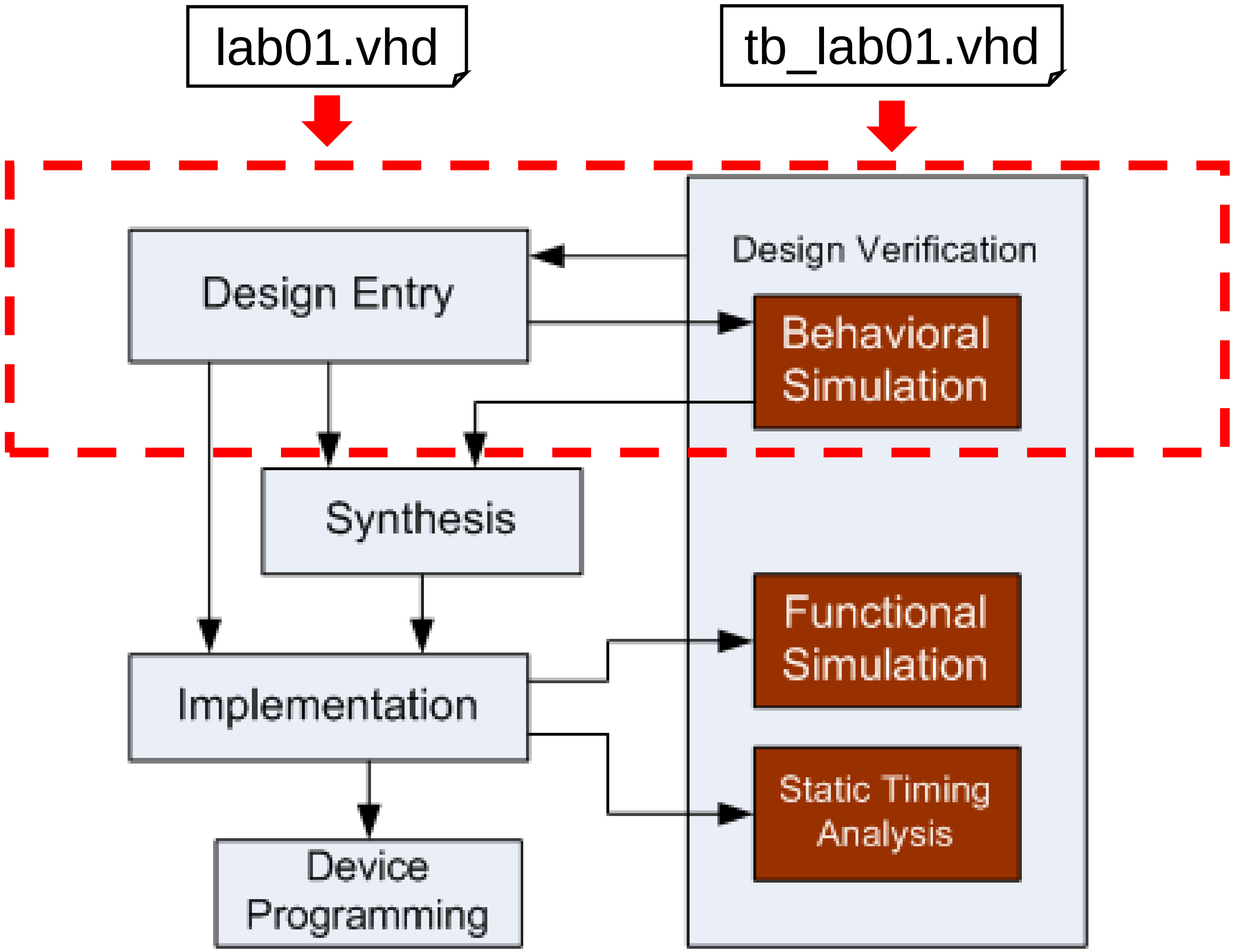
Timing

Timing information is not available because it hasn't been run

Design Runs

Name	Part	Constraints	Strategy	Status	Progress	Start	Elapsed	Failed Routes	WNS	TNS	WHS	THS	TPWS
synth_1	xc7z020clg484-1	constrs_1	Vivado Synthesis Defaults (Vivado Synthesis 2013)	synth_design Complete!	100%	12/29/15 4:17 PM	00:00:40						
impl_1	xc7z020clg484-1	constrs_1	Vivado Implementation Defaults (Vivado Implementation 2013)	write_bitstream Complete!	100%	12/30/15 11:16 AM	00:01:31	0	0.000	0.000	0.000	0.000	0.000

Focus of Lab01: Behavioral Simulation



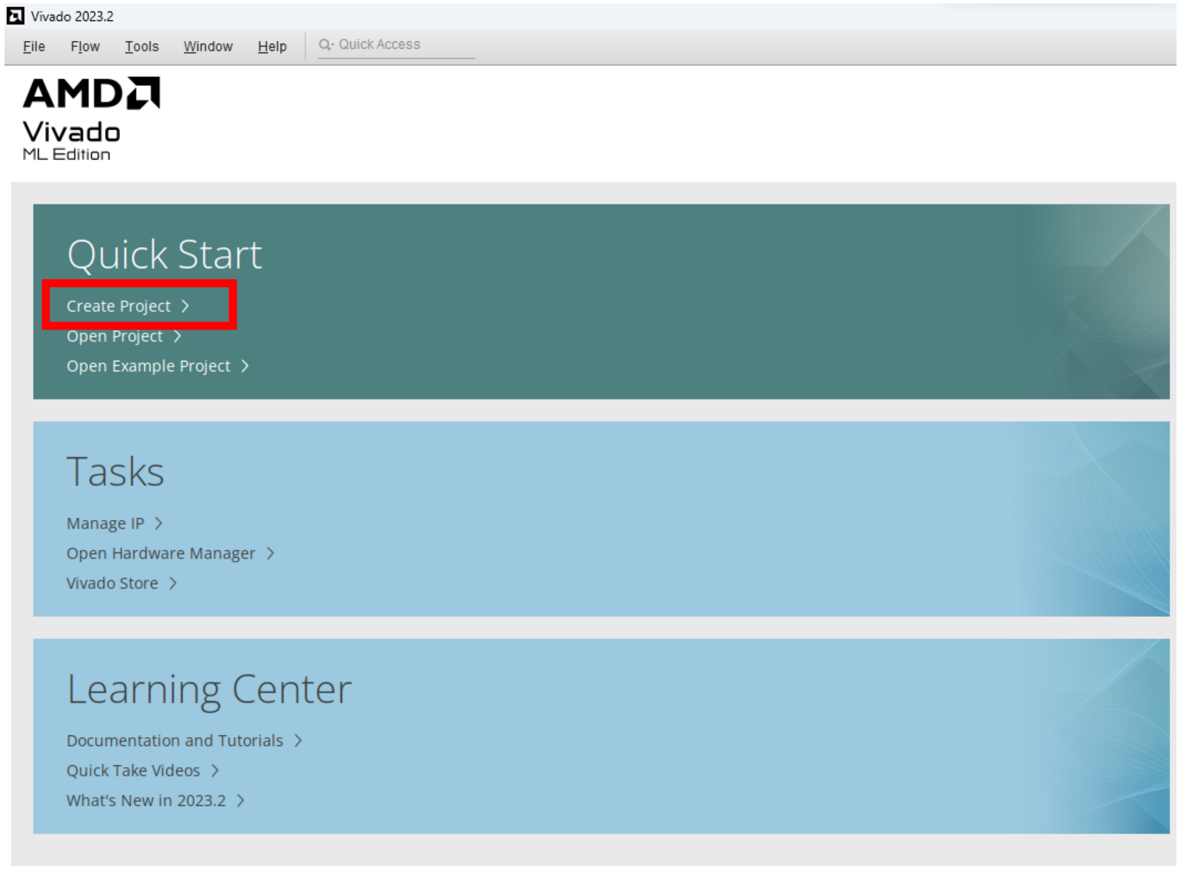
I. Create a Vivado Project

Step 1: Start the Software

- Double click the icon of “Vivado”
- Or start “Vivado” from the Start Menu

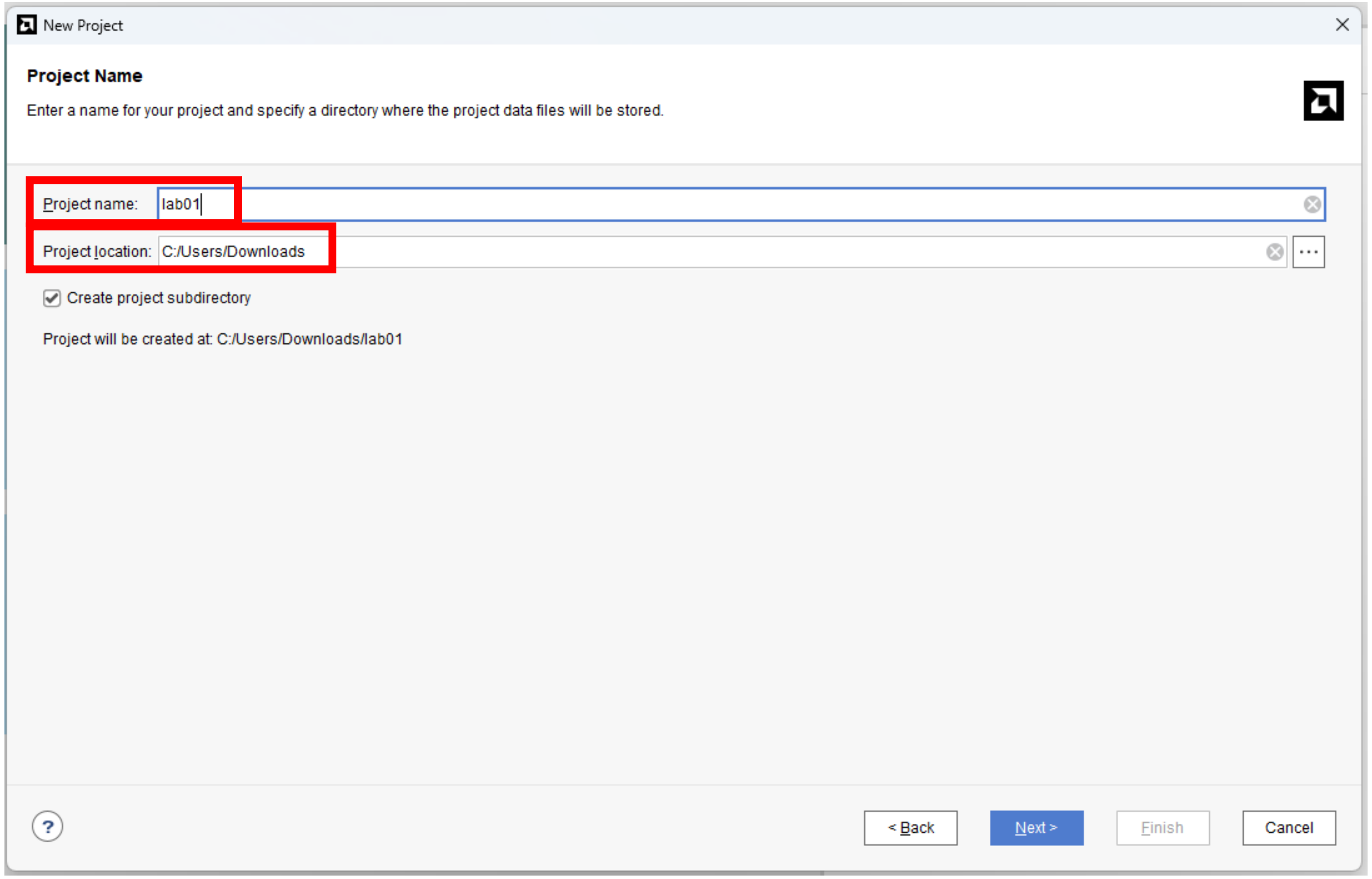
Step 2: Create a New Project

- Click “Create Project”
- Or click “File” -> “Project” -> “New”



Step 2: Create a New Project

- Enter the “Project name” (e.g., lab01)
- Specify a proper “Project location” for storage



New Project

Project Name

Enter a name for your project and specify a directory where the project data files will be stored.

Project name: lab01

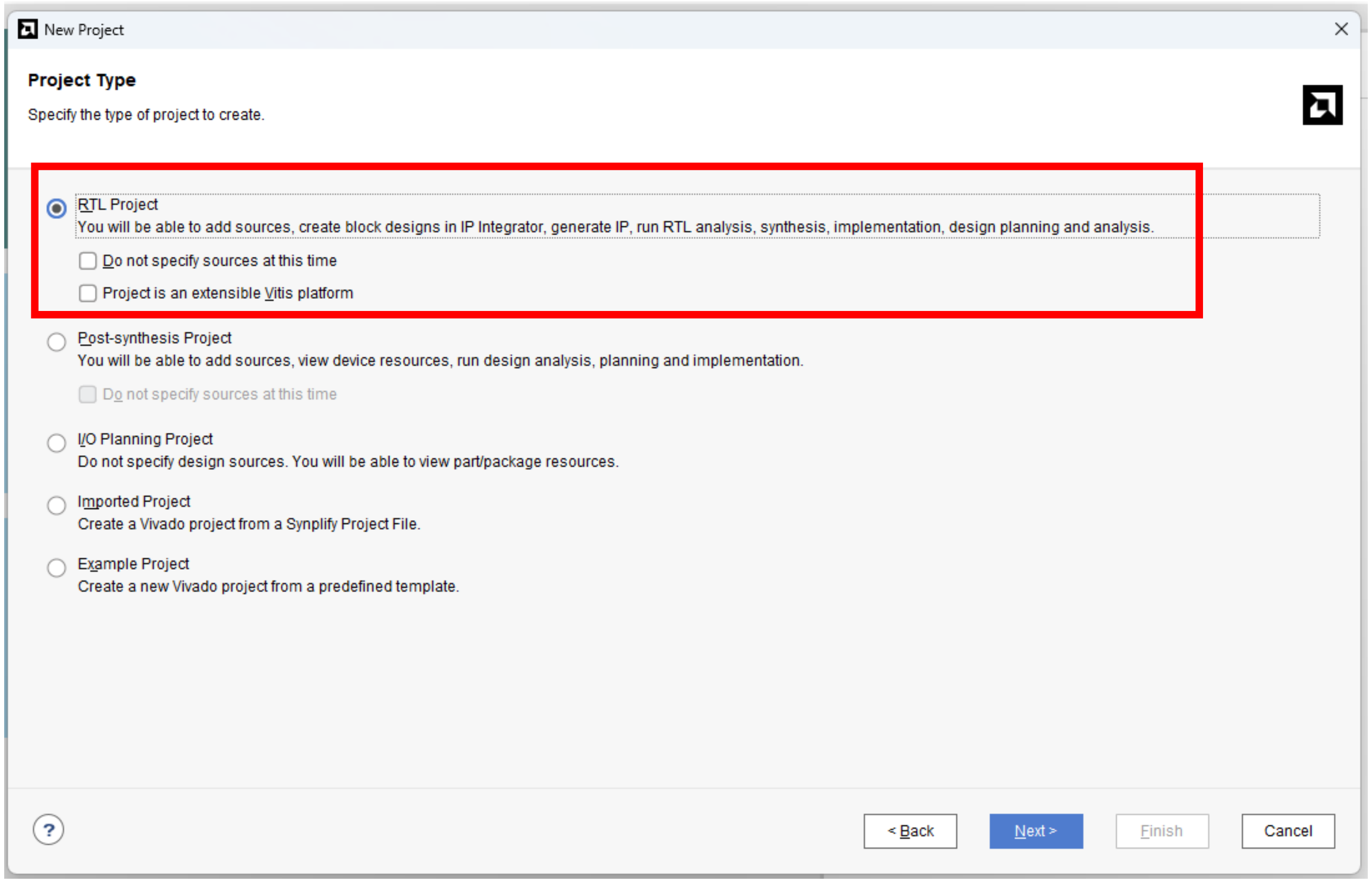
Project location: C:/Users/Downloads

☑ Create project subdirectory

Project will be created at: C:/Users/Downloads/lab01

Step 2: Create a New Project

- Select “RTL Project” -> “Next”



New Project

Project Type

Specify the type of project to create.

[✓ RTL Project]
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.

☐ Do not specify sources at this time

☐ Project is an extensible Vitis platform

Post-synthesis Project
You will be able to add sources, view device resources, run design analysis, planning and implementation.

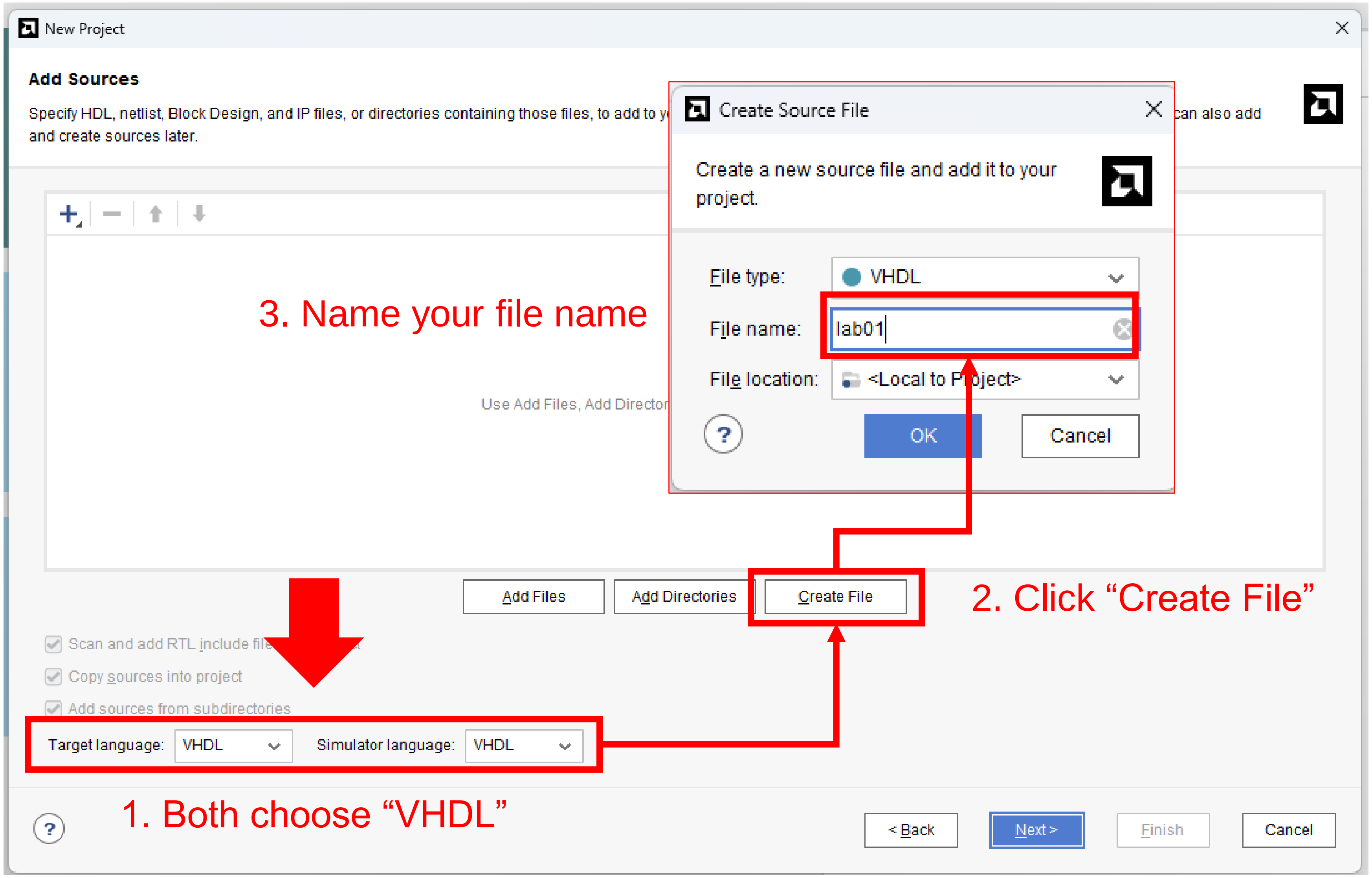
☐ Do not specify sources at this time

I/O Planning Project
Do not specify design sources. You will be able to view part/package resources.

Imported Project
Create a Vivado project from a Synplify Project File.

Example Project
Create a new Vivado project from a predefined template.

Step 2: Create a New Project



New Project

Add Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those files, to add to y… can also add and create sources later.

Use Add Files, Add Director…

Add Files | Add Directories | Create File

- [✓] Scan and add RTL include file…
- [✓] Copy sources into project
- [✓] Add sources from subdirectories

Target language: VHDL   Simulator language: VHDL

< Back | Next > | Finish | Cancel

Create Source File

Create a new source file and add it to your project.

File type: VHDL

File name: lab01

File location: <Local to Project>

OK | Cancel

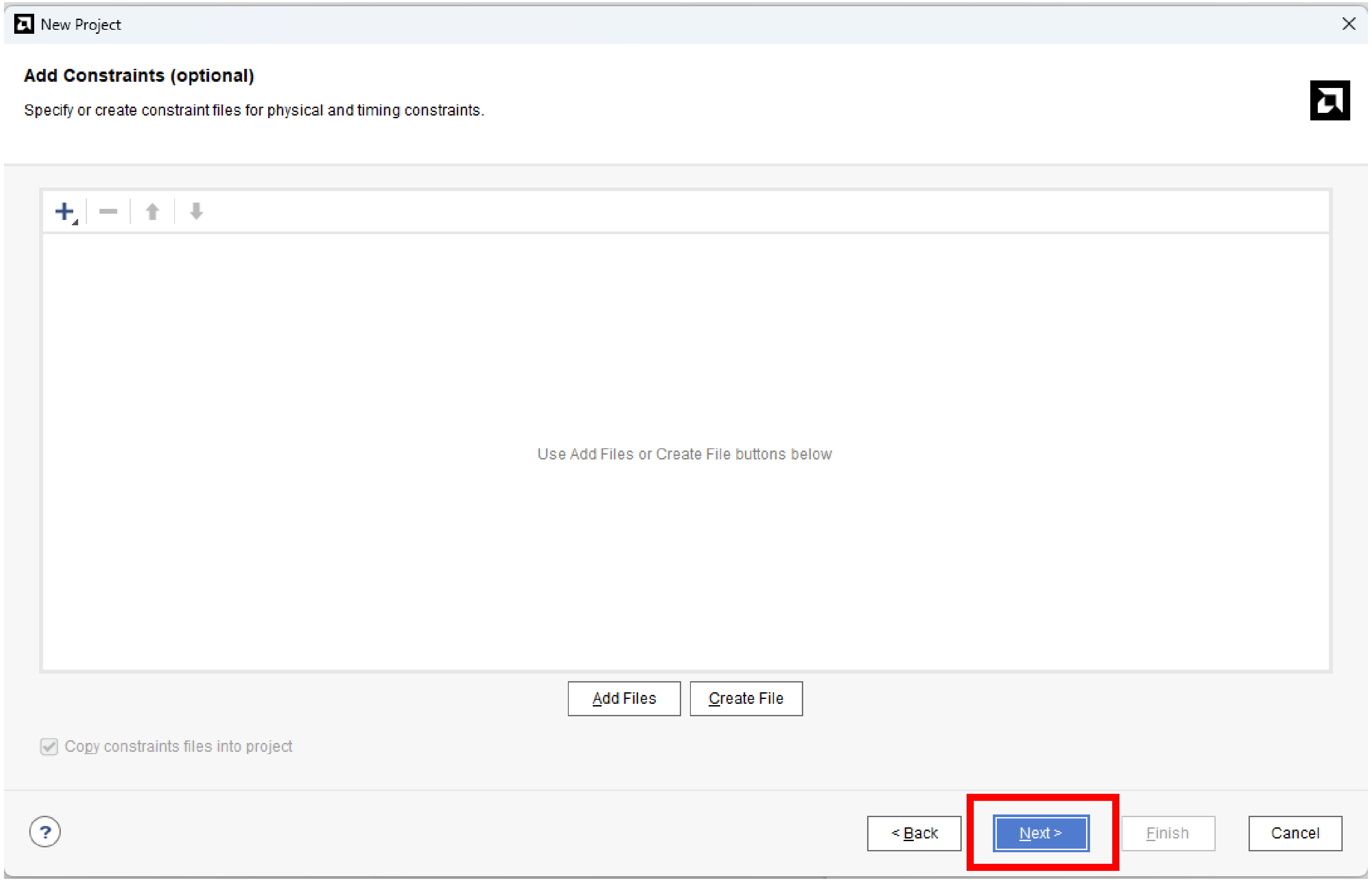
3. Name your file name

2. Click “Create File”

1. Both choose “VHDL”

Step 2: Create a New Project

- Click “Next”


New Project

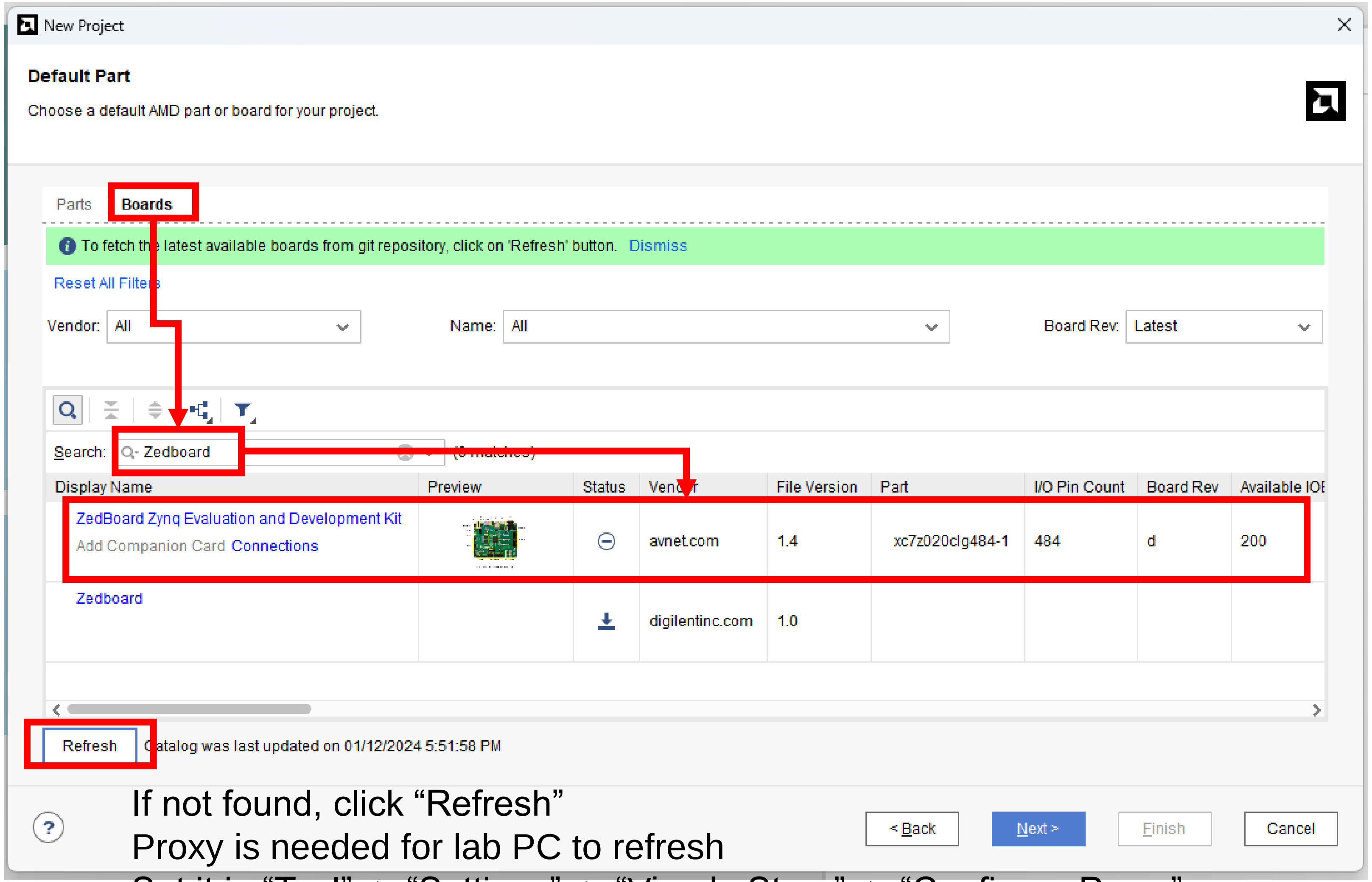
Add Constraints (optional)

Specify or create constraint files for physical and timing constraints.

Use Add Files or Create File buttons below

Step 2: Create a New Project

- Click "Boards" -> Select "ZedBoard" -> Click "Next"



New Project

Default Part

Choose a default AMD part or board for your project.

To fetch the latest available boards from git repository, click on 'Refresh' button. Dismiss

Reset All Filters

Vendor: All

Name: All

Board Rev: Latest

Search: Zedboard

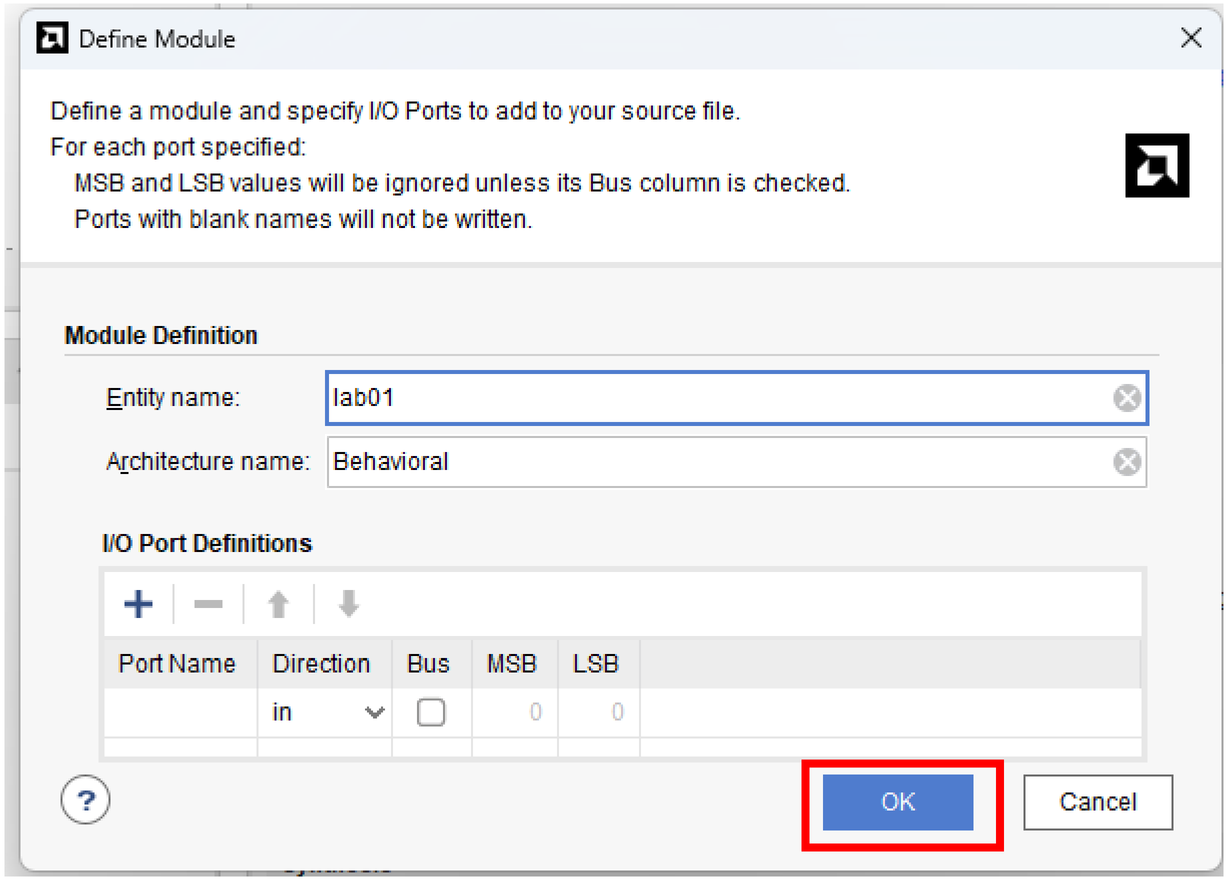
Display Name	Preview	Status	Vendor	File Version	Part	I/O Pin Count	Board Rev	Available IOB
ZedBoard Zynq Evaluation and Development Kit Add Companion Card Connections		⊖	avnet.com	1.4	xc7z020clg484-1	484	d	200
Zedboard		⤓	digilentinc.com	1.0				

Refresh Catalog was last updated on 01/12/2024 5:51:58 PM

If not found, click "Refresh"
Proxy is needed for lab PC to refresh
Set it in "Tool" -> "Settings" -> "Vivado Store" -> "Configure Proxy"

Step 2: Create a New Project

- If you see this box, just click "OK"


Define Module

Define a module and specify I/O Ports to add to your source file.
For each port specified:
MSB and LSB values will be ignored unless its Bus column is checked.
Ports with blank names will not be written.

Module Definition

Entity name: lab01

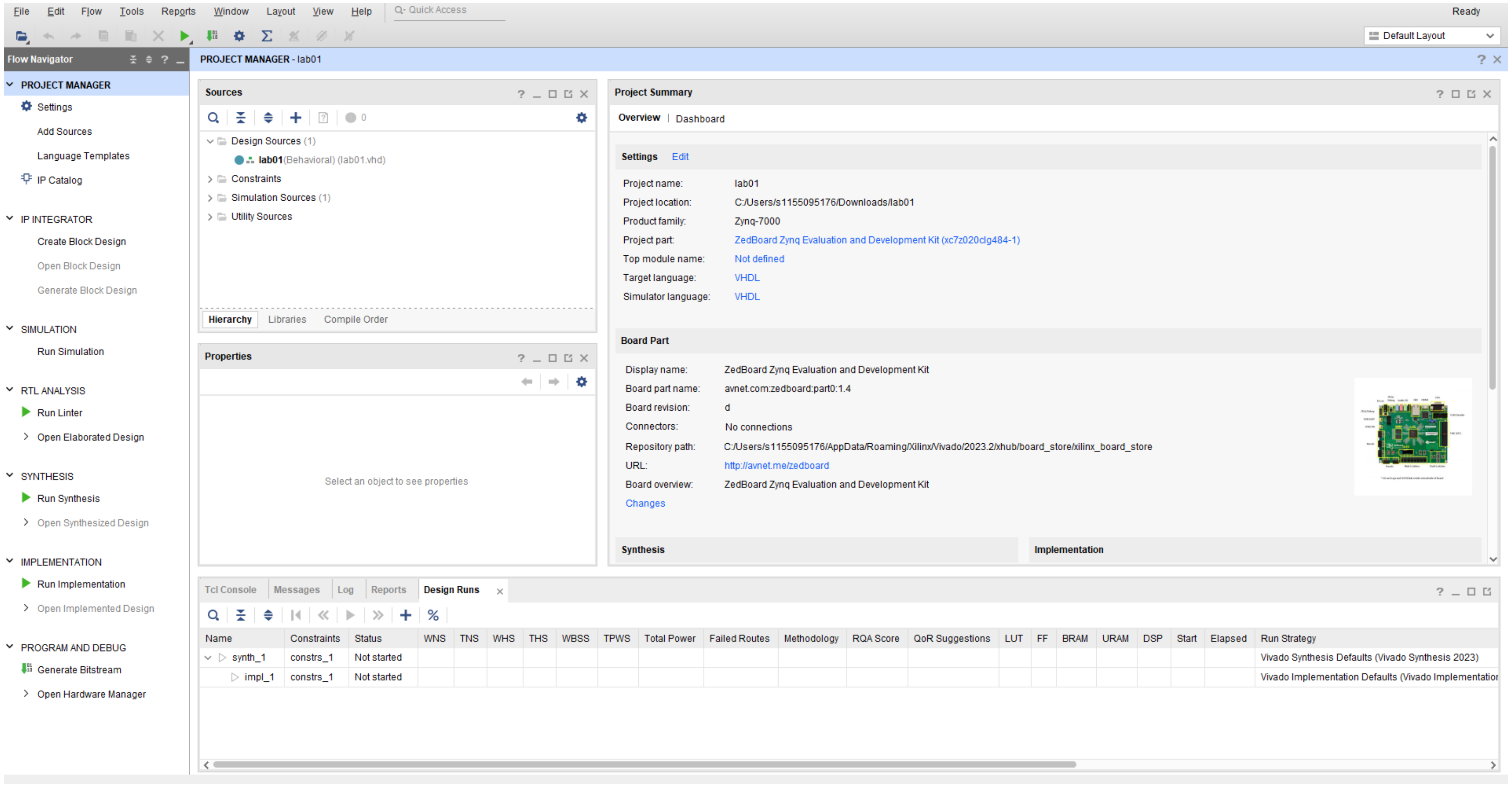
Architecture name: Behavioral

I/O Port Definitions

Port Name	Direction	Bus	MSB	LSB
	in	☐	0	0

Step 2: Create a New Project

- This is the programming interface of Vivado:



Settings Edit

Project name:	lab01
Project location:	C:/Users/s1155095176/Downloads/lab01
Product family:	Zynq-7000
Project part:	ZedBoard Zynq Evaluation and Development Kit (xc7z020clg484-1)
Top module name:	Not defined
Target language:	VHDL
Simulator language:	VHDL

Board Part

Display name:	ZedBoard Zynq Evaluation and Development Kit
Board part name:	avnet.com:zedboard:part0:1.4
Board revision:	d
Connectors:	No connections
Repository path:	C:/Users/s1155095176/AppData/Roaming/Xilinx/Vivado/2023.2/xhub/board_store/xilinx_board_store
URL:	http://avnet.me/zedboard
Board overview:	ZedBoard Zynq Evaluation and Development Kit

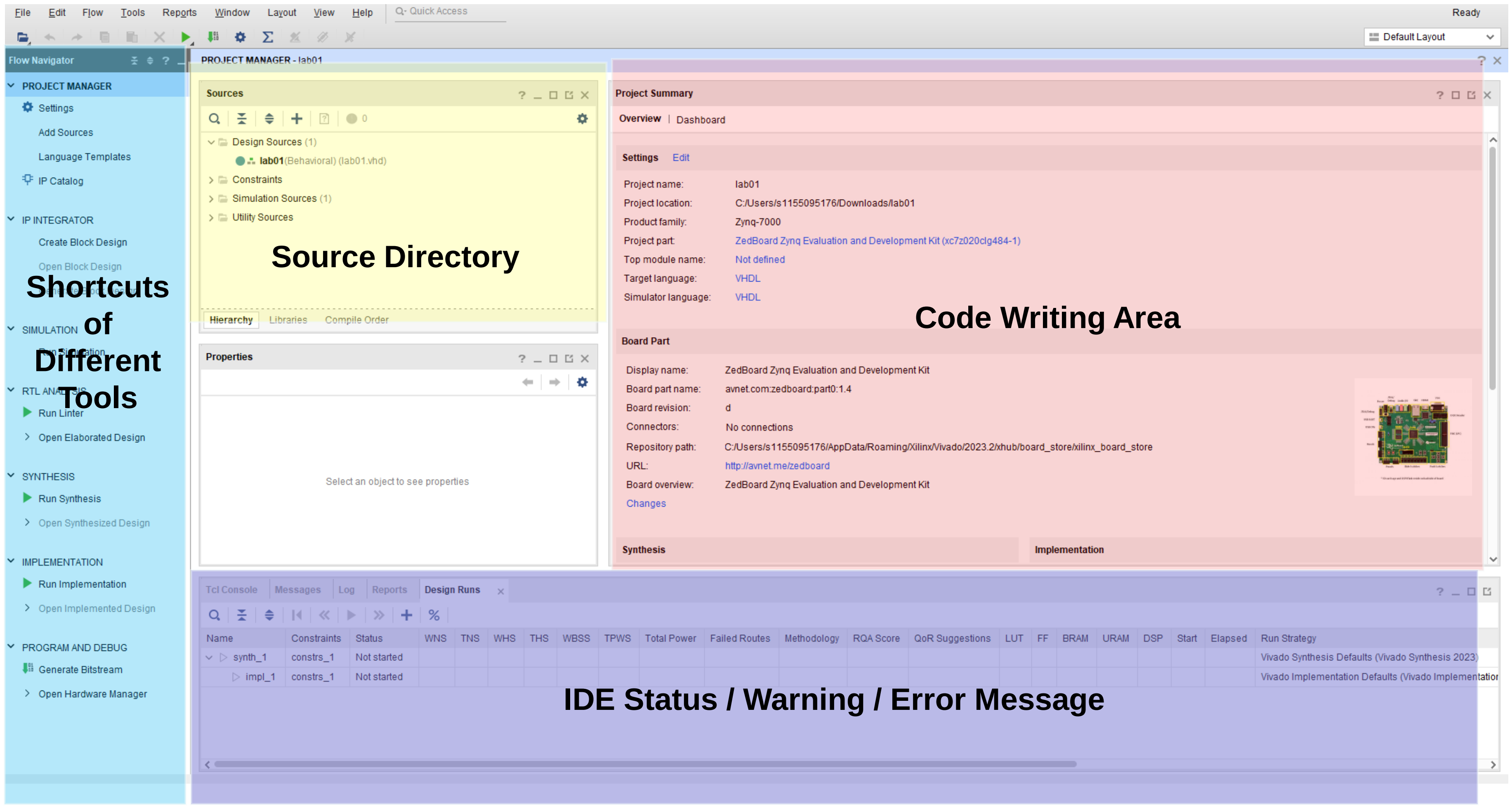
Changes

Synthesis **Implementation**

Name	Constraints	Status	WNS	TNS	WHS	THS	WBSS	TPWS	Total Power	Failed Routes	Methodology	RQA Score	QoR Suggestions	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Run Strategy
synth_1	constrs_1	Not started																			Vivado Synthesis Defaults (Vivado Synthesis 2023)
impl_1	constrs_1	Not started																			Vivado Implementation Defaults (Vivado Implementatior

II. Writing the VHDL Code

- This is the programming interface of Vivado:



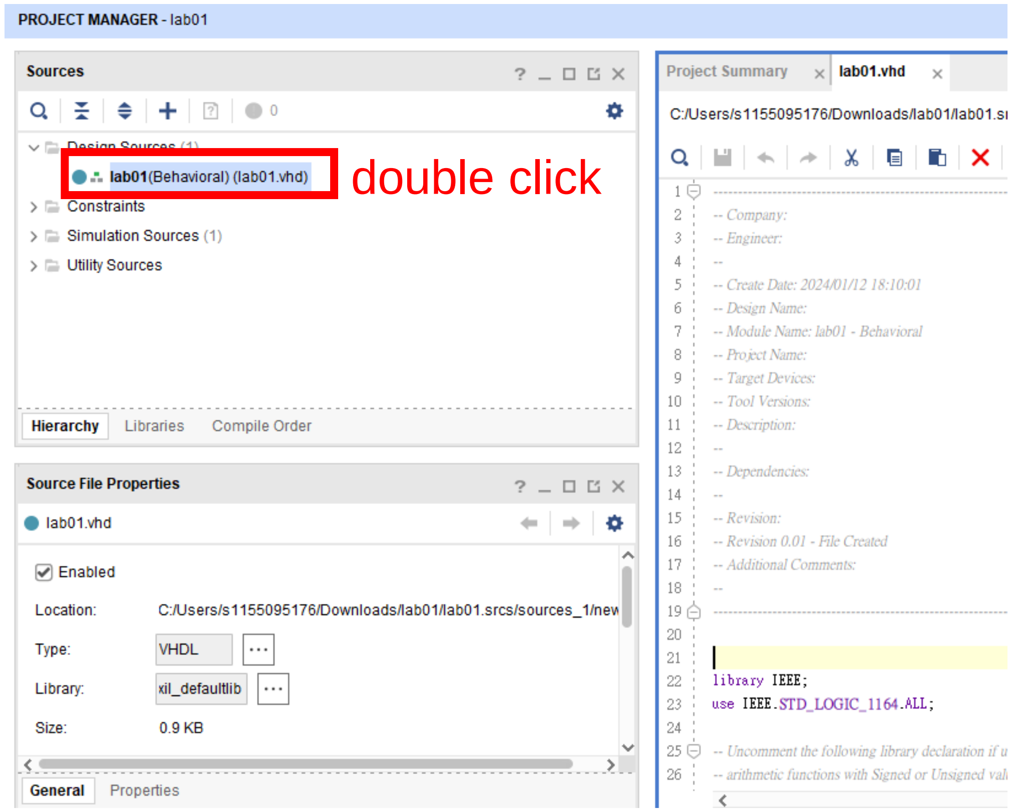
Project name:	lab01
Project location:	C:/Users/s1155095176/Downloads/lab01
Product family:	Zynq-7000
Project part:	ZedBoard Zynq Evaluation and Development Kit (xc7z020clg484-1)
Top module name:	Not defined
Target language:	VHDL
Simulator language:	VHDL

Display name:	ZedBoard Zynq Evaluation and Development Kit
Board part name:	avnet.com:zedboard:part0:1.4
Board revision:	d
Connectors:	No connections
Repository path:	C:/Users/s1155095176/AppData/Roaming/Xilinx/Vivado/2023.2/xhub/board_store/xilinx_board_store
URL:	http://avnet.me/zedboard
Board overview:	ZedBoard Zynq Evaluation and Development Kit

Name	Constraints	Status	WNS	TNS	WHS	THS	WBSS	TPWS	Total Power	Failed Routes	Methodology	RQA Score	QoR Suggestions	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Run Strategy
synth_1	constrs_1	Not started																			Vivado Synthesis Defaults (Vivado Synthesis 2023)
impl_1	constrs_1	Not started																			Vivado Implementation Defaults (Vivado Implementatior

Step 3: Start Programming

- In the “Source Directory”, double click the source file (e.g., lab01.vhd) created.
  - *Seeing the file name appears twice in “Syntax Error Files” and “Non-module Files” is normal if the source code is empty or having syntax errors.*



Enabled

Location: C:/Users/s1155095176/Downloads/lab01/lab01.srcs/sources_1/new

Type: VHDL

Library: xil_defaultlib

Size: 0.9 KB

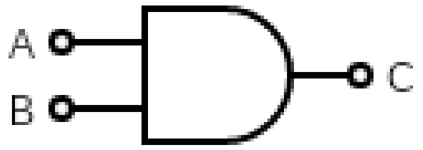
C:/Users/s1155095176/Downloads/lab01/lab01.s

```
----------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 2024/01/12 18:10:01
-- Design Name:
-- Module Name: lab01 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if u
-- arithmetic functions with Signed or Unsigned val
```

Step 3: Start Programming

- Implement a simple AND logic and save

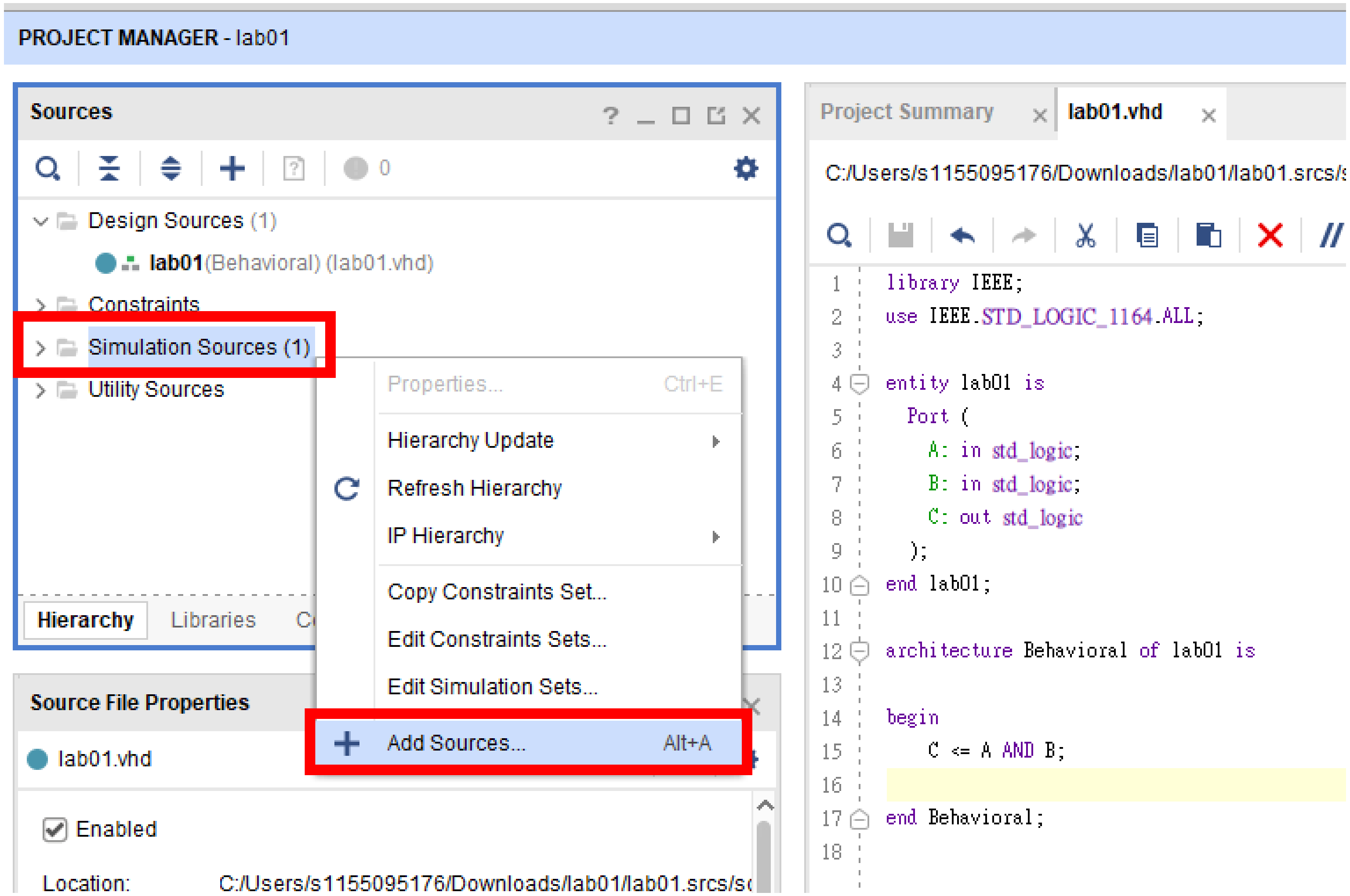
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity lab01 is
  Port (
    A: in std_logic;
    B: in std_logic;
    C: out std_logic
  );
end lab01;

architecture Behavioral of lab01 is

begin
    C <= A AND B;

end Behavioral;
```



III. Perform the Simulation

Step 4: Write a Testbench

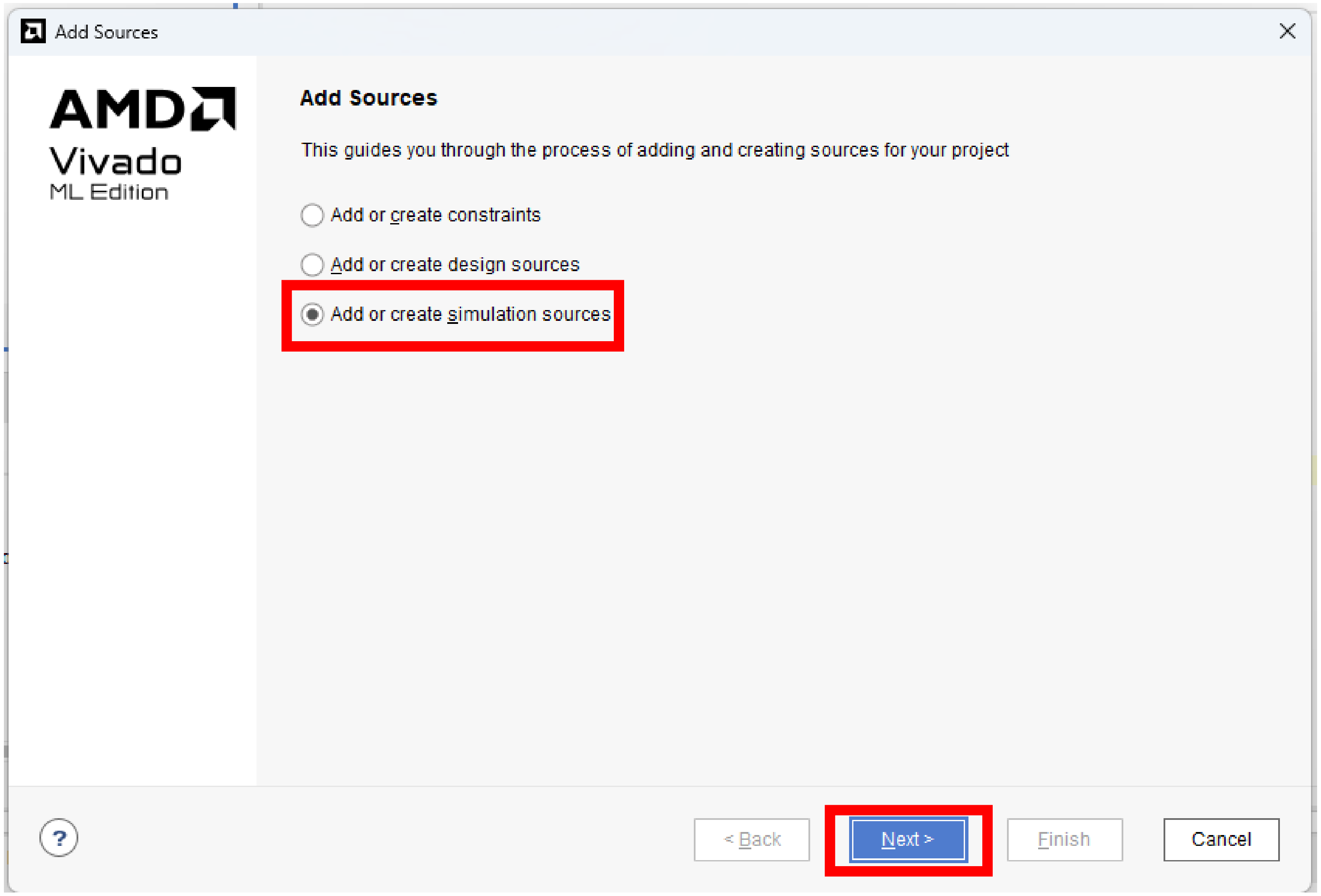
- Under the source window, right click on the “Simulation Sources” -> “Add sources…”



```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity lab01 is
  Port (
    A: in std_logic;
    B: in std_logic;
    C: out std_logic
  );
end lab01;

architecture Behavioral of lab01 is

begin
    C <= A AND B;

end Behavioral;

```

Step 4: Write a Testbench

- Select "Add or Create Simulation Sources" -> "Next"



Add Sources

Add Sources

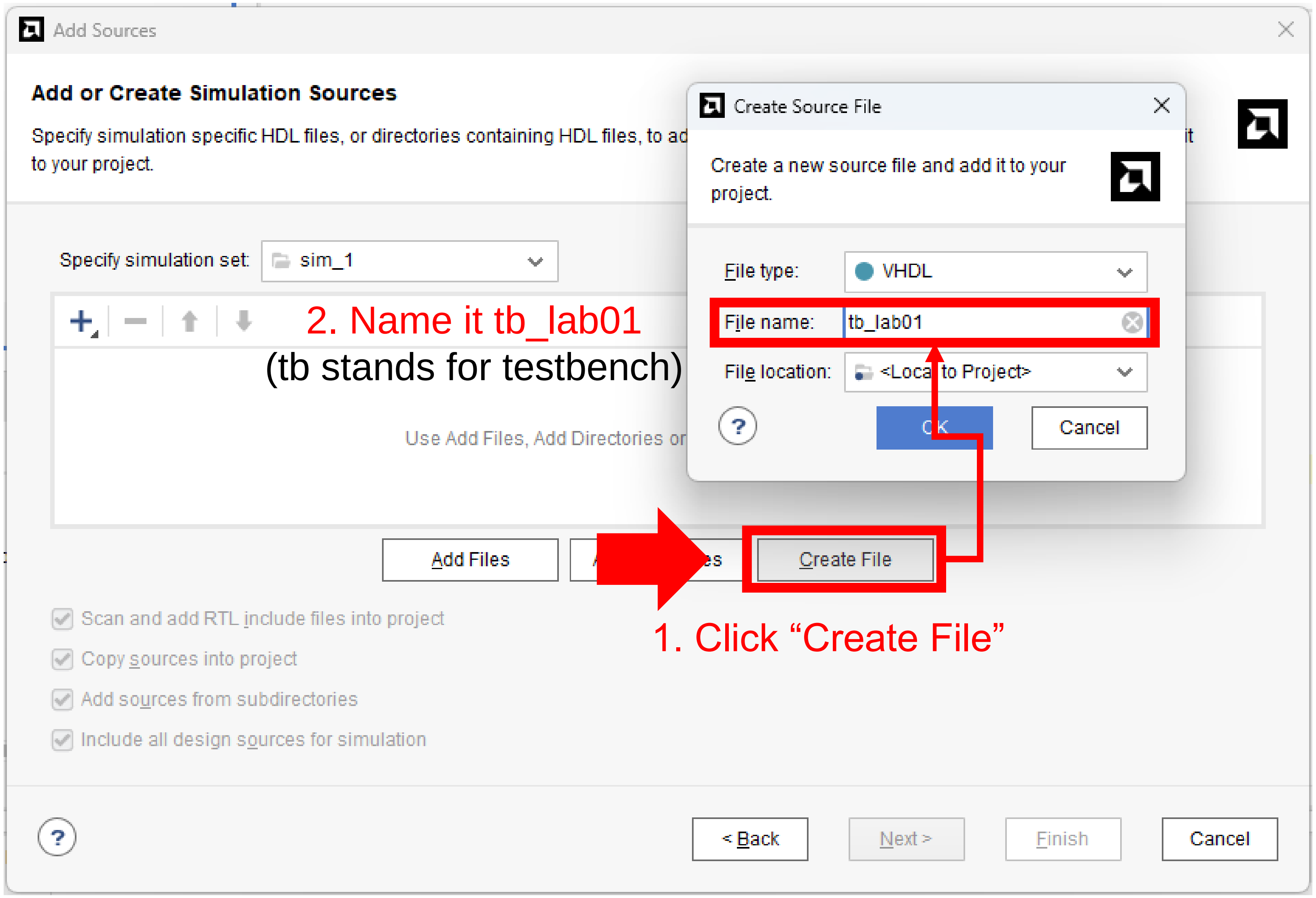
This guides you through the process of adding and creating sources for your project

○ Add or create constraints

○ Add or create design sources

● Add or create simulation sources

Step 4: Write a Testbench



Add Sources

Add or Create Simulation Sources

Specify simulation specific HDL files, or directories containing HDL files, to ad... it to your project.

Specify simulation set: sim_1

2. Name it tb_lab01
(tb stands for testbench)

Use Add Files, Add Directories or

Create Source File

Create a new source file and add it to your project.

File type: VHDL

File name: tb_lab01

File location: <Local to Project>

OK | Cancel

Add Files | Create File

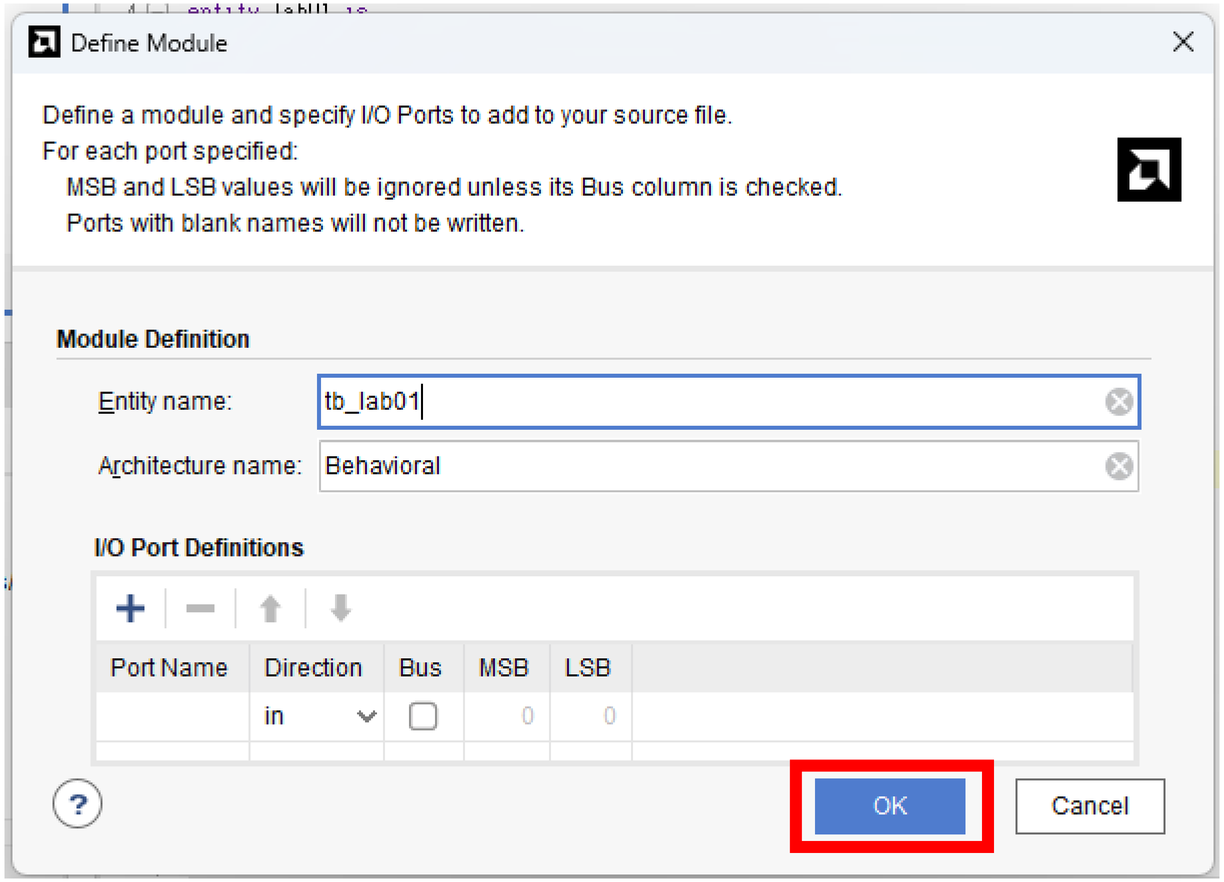
1. Click "Create File"

- ☑ Scan and add RTL include files into project
- ☑ Copy sources into project
- ☑ Add sources from subdirectories
- ☑ Include all design sources for simulation

< Back | Next > | Finish | Cancel

Step 4: Write a Testbench

- If you see this box, just click “OK”



Define Module

Define a module and specify I/O Ports to add to your source file.
For each port specified:
MSB and LSB values will be ignored unless its Bus column is checked.
Ports with blank names will not be written.

Module Definition

Entity name: tb_lab01

Architecture name: Behavioral

I/O Port Definitions

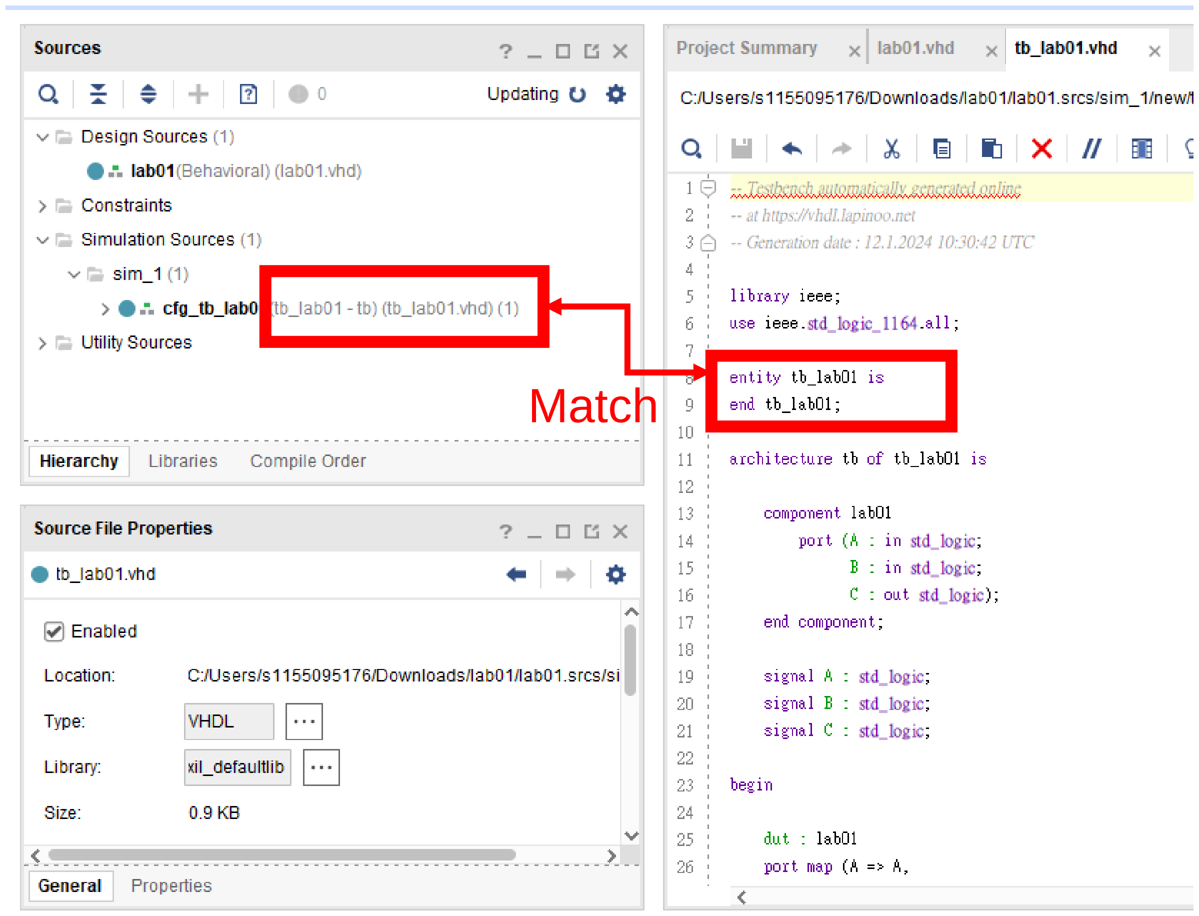
Port Name	Direction	Bus	MSB	LSB
	in	☐	0	0

Step 4: Write a Testbench

- The testbench file will not be generated automatically
- An easy way is to use Testbench Template Generator
  1. Visit https://vhdl.lapinoo.net/testbench/
  2. Paste your source code into it, and the website will generate the testbench file for you
     - ✓ Select “No clock generation”
     - ✓ Select “No reset generation”
  3. Copy the content generated by the website, and replace the content on tb_lab01.vhd

Step 4: Write a Testbench

- Make sure that the entity name (i.e., tb_lab0) in the testbench file matches the testbench file name



Location: C:/Users/s1155095176/Downloads/lab01/lab01.srcs/si

Type: VHDL

Library: xil_defaultlib

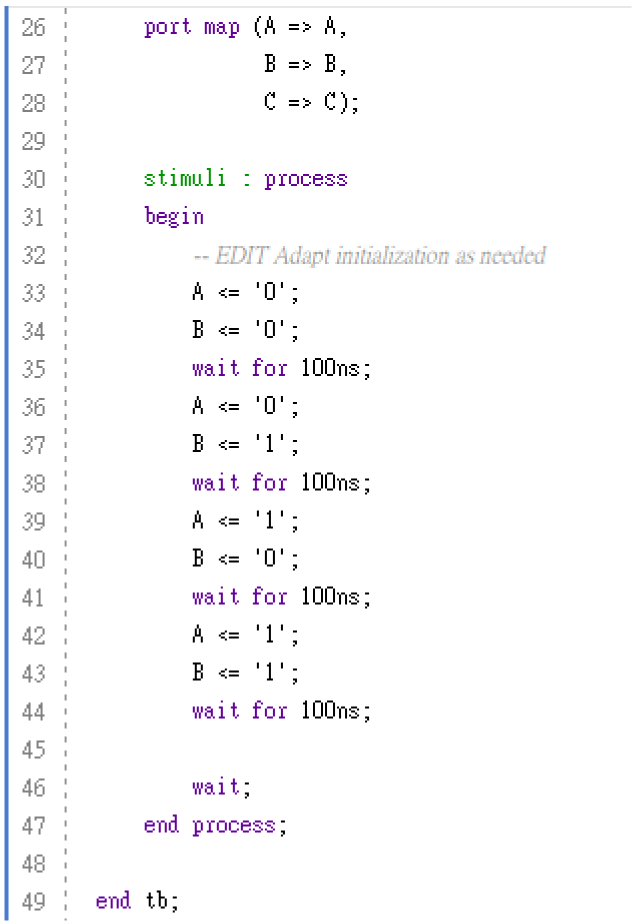
Size: 0.9 KB

C:/Users/s1155095176/Downloads/lab01/lab01.srcs/sim_1/new/t

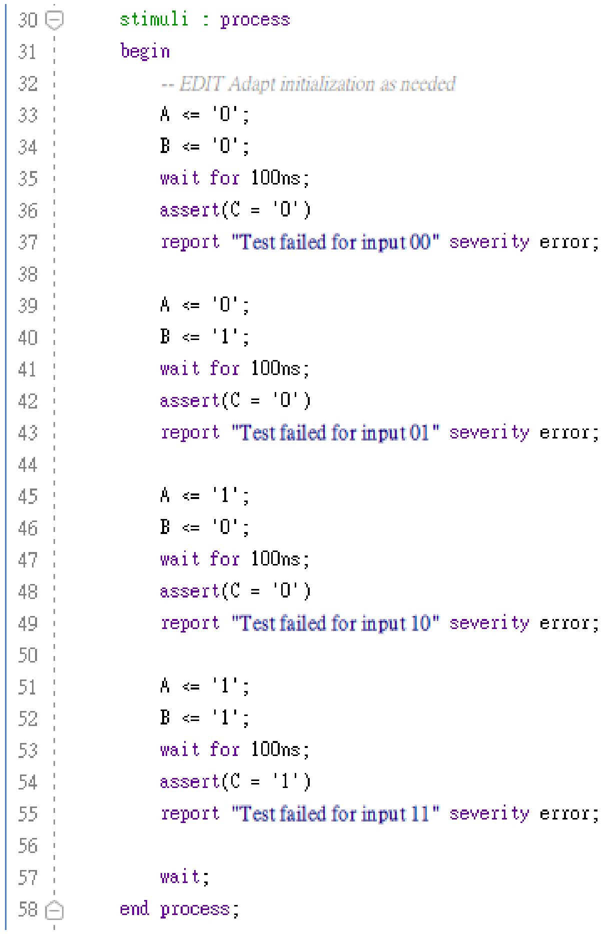
```
-- Testbench automatically generated online
-- at https://vhdl.lapinoo.net
-- Generation date : 12.1.2024 10:30:42 UTC

library ieee;
use ieee.std_logic_1164.all;

entity tb_lab01 is
end tb_lab01;

architecture tb of tb_lab01 is

    component lab01
        port (A : in std_logic;
              B : in std_logic;
              C : out std_logic);
    end component;

    signal A : std_logic;
    signal B : std_logic;
    signal C : std_logic;

begin

    dut : lab01
    port map (A => A,
```

Step 4: Write a Testbench

- Finish the *stimuli* process with one style below



```vhdl
    port map (A => A,
              B => B,
              C => C);

    stimuli : process
    begin
        -- EDIT Adapt initialization as needed
        A <= '0';
        B <= '0';
        wait for 100ns;
        A <= '0';
        B <= '1';
        wait for 100ns;
        A <= '1';
        B <= '0';
        wait for 100ns;
        A <= '1';
        B <= '1';
        wait for 100ns;

        wait;
    end process;

end tb;
```



```vhdl
    stimuli : process
    begin
        -- EDIT Adapt initialization as needed
        A <= '0';
        B <= '0';
        wait for 100ns;
        assert(C = '0')
        report "Test failed for input 00" severity error;

        A <= '0';
        B <= '1';
        wait for 100ns;
        assert(C = '0')
        report "Test failed for input 01" severity error;

        A <= '1';
        B <= '0';
        wait for 100ns;
        assert(C = '0')
        report "Test failed for input 10" severity error;

        A <= '1';
        B <= '1';
        wait for 100ns;
        assert(C = '1')
        report "Test failed for input 11" severity error;

        wait;
    end process;
```

Style 1 (Simpler)
(copyable version on the next slide)

Style 2
(With assertion)

Step 4: Write a Testbench

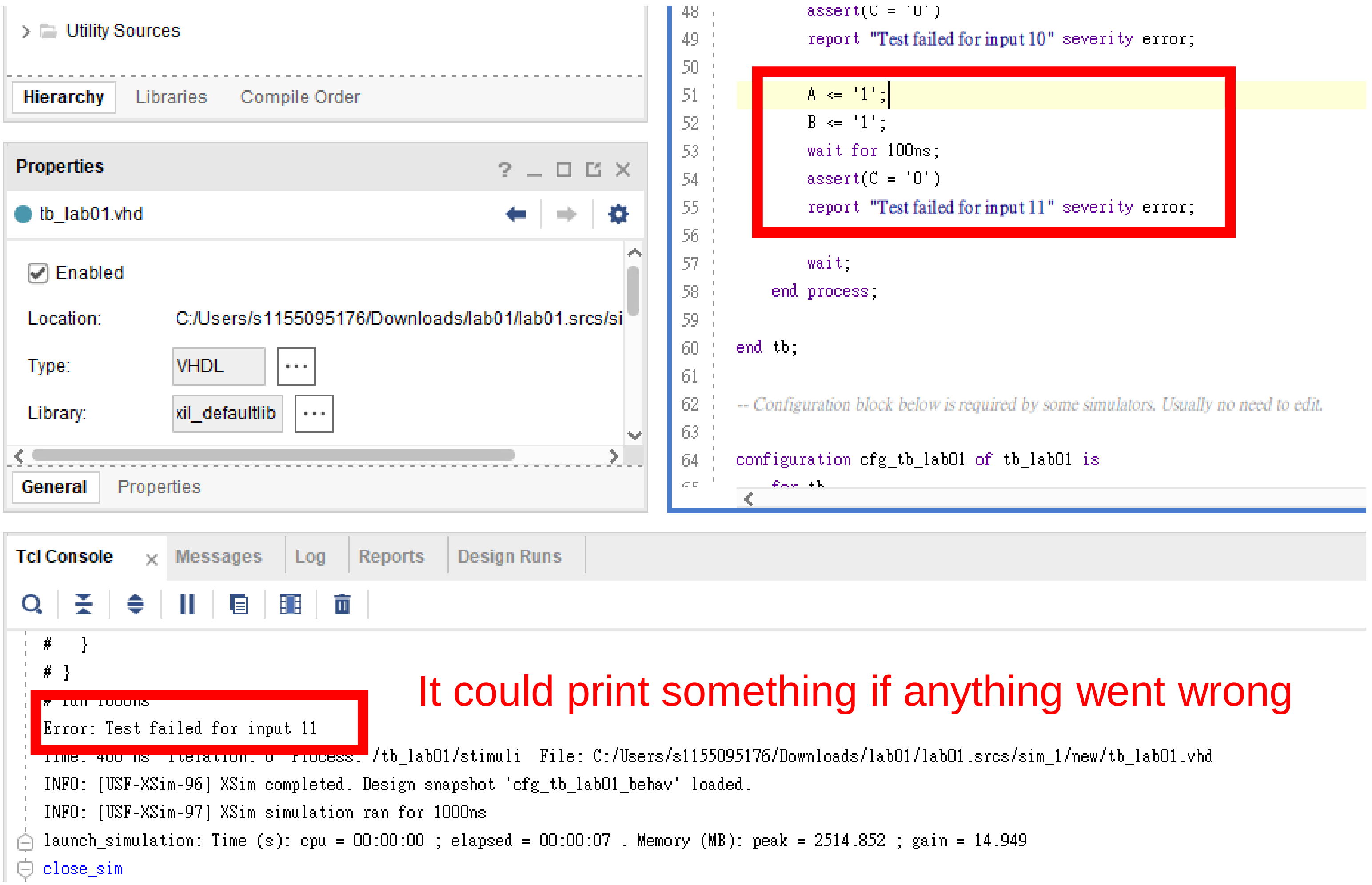
- Testbench Style 1

```
stimuli : process
    begin
        A <= '0';
        B <= '0';
        wait for 100ns;
        A <= '0';
        B <= '1';
        wait for 100ns;
        A <= '1';
        B <= '0';
        wait for 100ns;
        A <= '1';
        B <= '1';
        wait for 100ns;
        wait;
    end process;
```

You can copy this

Step 4: Write a Testbench

- Why Style 2 ?



tb_lab01.vhd

☑ Enabled

Location: C:/Users/s1155095176/Downloads/lab01/lab01.srcs/si

Type: VHDL

Library: xil_defaultlib

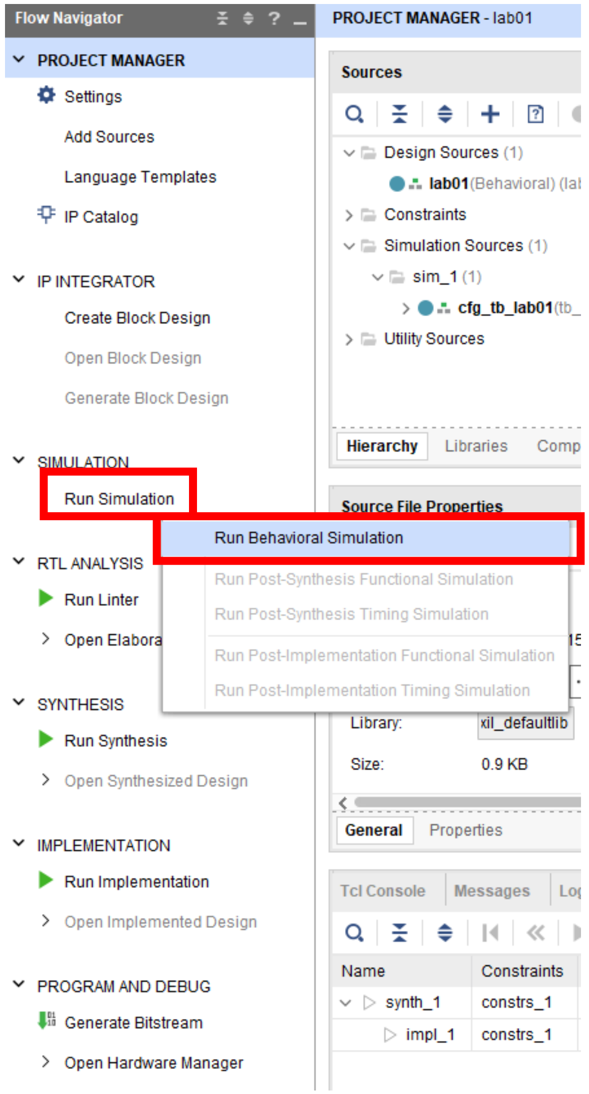
```
            assert(C = '0')
            report "Test failed for input 10" severity error;

            A <= '1';
            B <= '1';
            wait for 100ns;
            assert(C = '0')
            report "Test failed for input 11" severity error;

            wait;
        end process;

end tb;

-- Configuration block below is required by some simulators. Usually no need to edit.

configuration cfg_tb_lab01 of tb_lab01 is
```

```
#   }
# }
# run 1000ns
Error: Test failed for input 11
Time: 400 ns  Iteration: 0  Process: /tb_lab01/stimuli  File: C:/Users/s1155095176/Downloads/lab01/lab01.srcs/sim_1/new/tb_lab01.vhd
INFO: [USF-XSim-96] XSim completed. Design snapshot 'cfg_tb_lab01_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:00 ; elapsed = 00:00:07 . Memory (MB): peak = 2514.852 ; gain = 14.949
close_sim
```

It could print something if anything went wrong

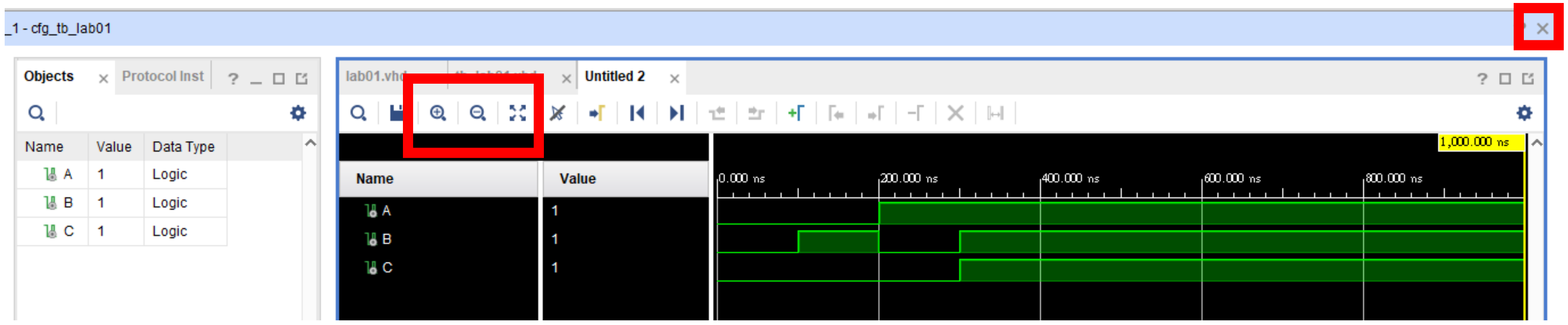
Step 5: Run Simulation

- Click “Run simulation” and select “Run Behavioral Simulation”



Step 5: Run Simulation

- Verify the output(s) by checking all possible input(s)
  - Click first, and use to adjust the timeline
  - If everything is alright, click X to exit



Name	Value	Data Type
A	1	Logic
B	1	Logic
C	1	Logic

Name	Value
A	1
B	1
C	1

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

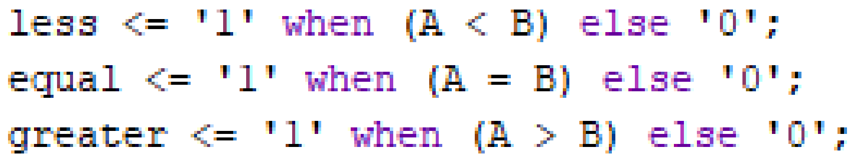
IV. Lab01 Task

Lab01: 2-bit Comparator

- **Objectives and Aims:**
  - Get familiar with VHDL and Vivado
- **Requirements:**
  1. Create a new project named “lab01”
  2. Implement a **2-bit comparator** using VHDL
     - 3 outputs are required (Both are std_logic)
       - **less**: When A < B, the output would be 1, otherwise 0
       - **equal**: When A = B, the output would be 1, otherwise 0
       - **greater**: When A > B, the output would be 1, otherwise 0
     - Hint: Refer the 4-bit comparator in Lec01 (Page 31)
  3. Write a **testbench** file to test **all possible** input(s)
  4. Run the **behavioral simulation** and screen capture the resulting waveform

Hint

- **For 2.**
  One possible solution is using when else syntax



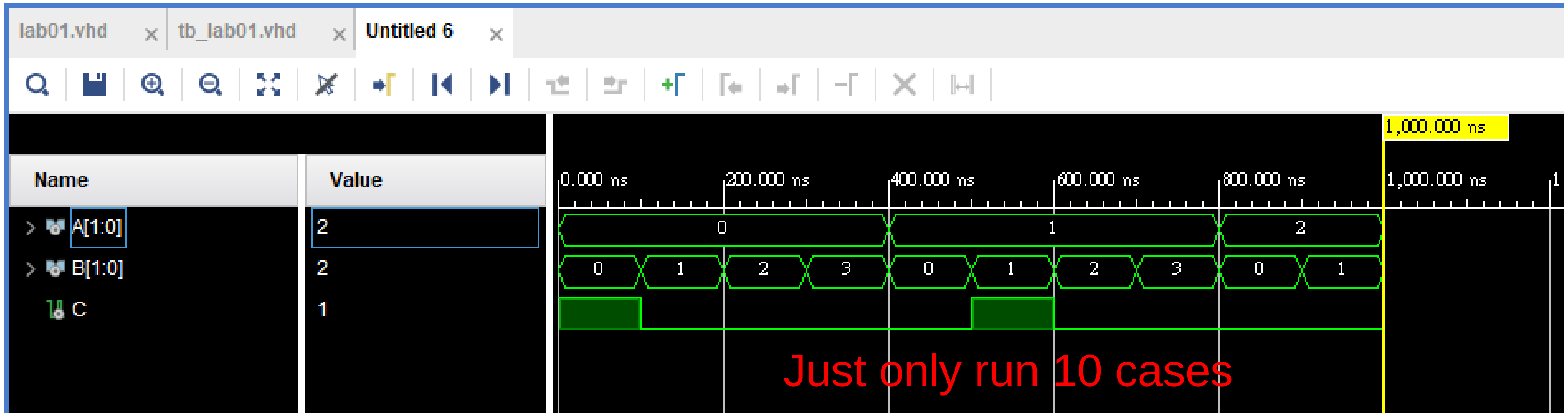
```
less <= '1' when (A < B) else '0';
equal <= '1' when (A = B) else '0';
greater <= '1' when (A > B) else '0';
```

- **For 3.**
  You can check all test cases with an Excel spreadsheet

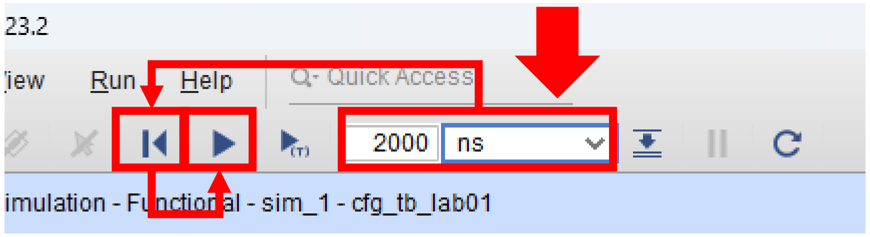
	A <= "00";	A <= "01";	A <= "10";	A <= "11";
B <= "00";				
B <= "01";				
B <= "10";				
B <= "11";				

Lab01: 2-bit Comparator

- One possible problem: The simulation might stop earlier than you expect

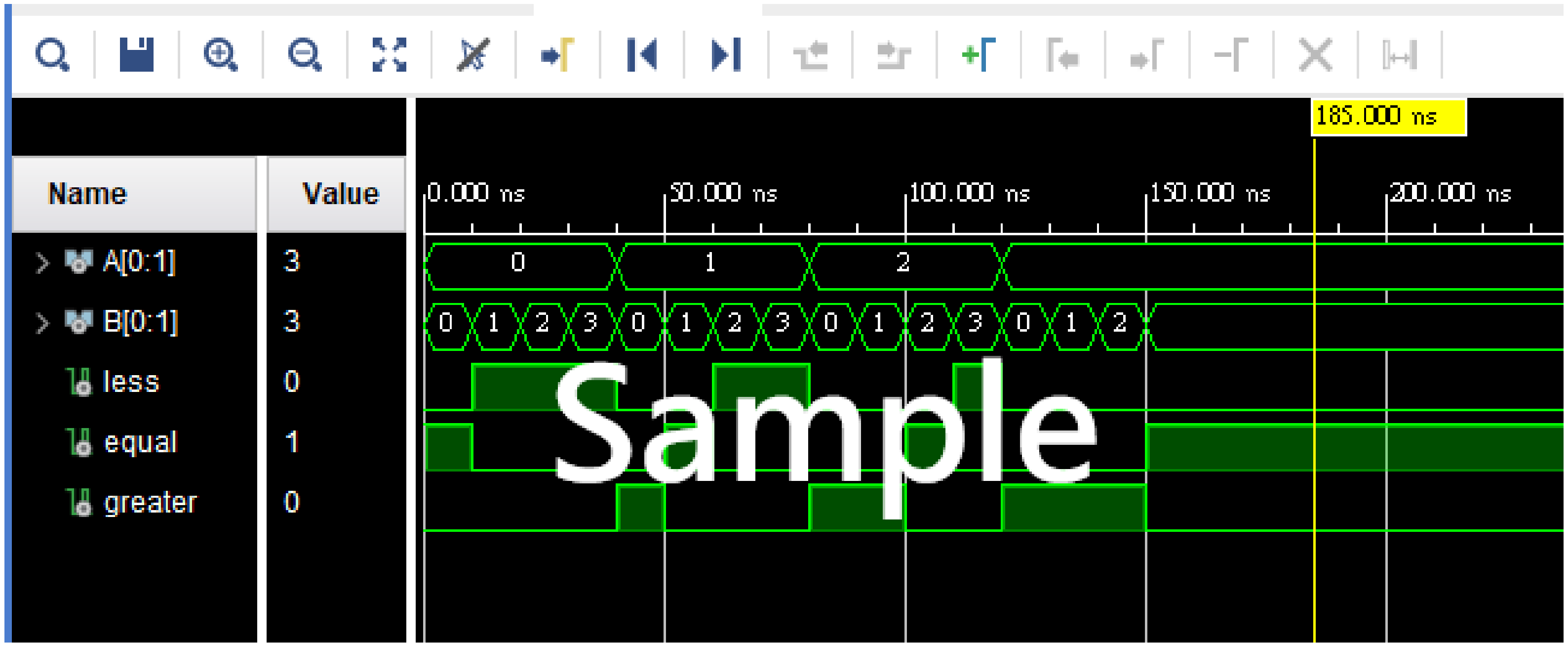


- Solution 1: Shorter the wait time (wait for 100ns -> 10ns)
- Solution 2:
  - Change to a longer duration
  - Click “Restart” then “Run All”

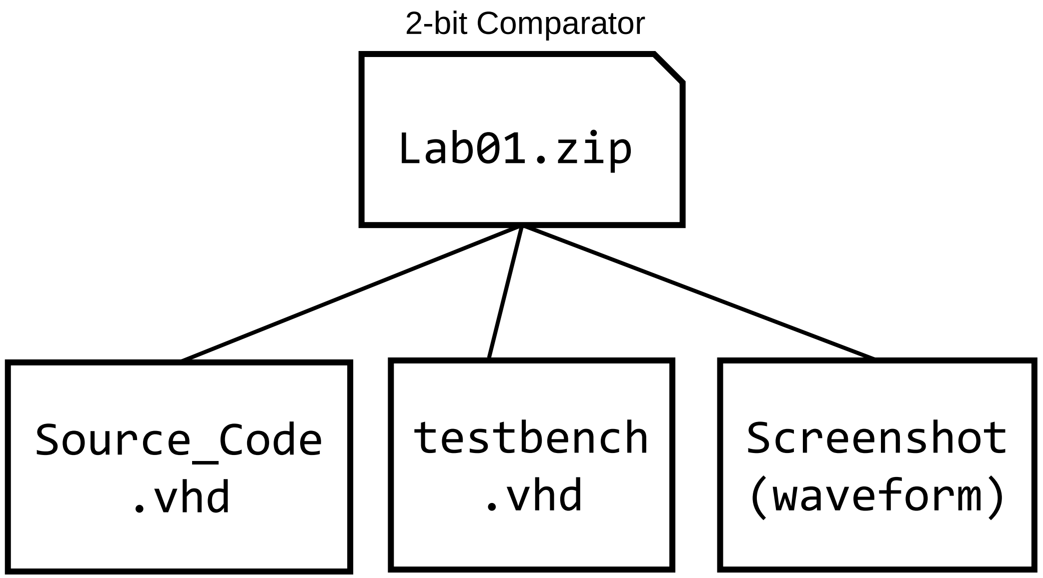


Lab01: 2-bit Comparator

- A sample simulation result of 2-bit comparator



Lab01: Submission File Checklist



Where to find the .vhd?

```
{Project Folder}
➔{Project Name}.srcs
        ➔sources_1
        ➔new
        ➔{source_code}.vhd
```
And
```
        ➔sim_1
        ➔new
        ➔{testbench}.vhd
```

- **Submission Rule:**
  1. Submit the zip file following the above format to Blackboard
     Deadline: **12:30 on 22 Jan. 2025**
     - ➢ Late submission is NOT acceptable (unless otherwise approved)

香港中文大學
The Chinese University of Hong Kong

Thank You!