

**CSE 490/590 COMPUTER ARCHITECTURE – SPRING 2019**  
**PROJECT 1 – DESIGN OF AN 8-BIT NON-PIPELINED PROCESSOR USING**  
**VERILOG**

**DUE : 15 APRIL 2019**

**PART 1**

This is a group project. Each group can have 2 students. A final project report is due on 15 April 2019. Groups should have both students from CSE 490 or both students from CSE 590.

There must be equal contribution from both students of the group. Individual contributions and specific tasks will be checked in the final project report and project demo.

Design an 8-bit microprocessor using Verilog HDL by using Structural Verilog modelling. The individual components can be designed using behavioral modelling.

The 8-bit **instruction formats** for R-type, I-type and J-type instructions are given below:

R-type instruction:

opcode			Rt/Rd	Rs	unused		
7	6	5	4	3	2	1	0

I-type instruction:

opcode			Rd	Rs	immediate		
7	6	5	4	3	2	1	0

J-type instruction:

opcode			address				
7	6	5	4	3	2	1	0

**Mandatory components:**

- Instruction Memory
- Register File
- Data Memory
- ALU
- Control Unit
- Multiplexers
- Sign extend unit
- Program counter

The Register File has two registers R0 and R1. Design the program counter and instruction memory such that input can be a sequence of instructions and the final output after executing the entire sequence can be verified.

Other than these components, there might be other modules needed depending on your design.

You can refer to the construction and working of the MIPS datapath and control path found in the text book “Computer Organization and Design – The Hardware/Software Interface” by Patterson and Hennessy, 4<sup>th</sup> ed. Also refer to the lecture slides on single cycle non-pipelined processor.

Chapter 4 in the above mentioned text book talks about building a datapath and a control path.

The instructions that you are required to implement are:

- i. lw
- ii. sw
- iii. add
- iv. addi
- v. jmp
- vi. sub

Instruction	Opcode
lw	000
sw	001
add	010
addi	011
sub	100
jmp	101

## PART 2

### Prerequisites:

The complete RTL schematic of the processor including the datapath and the control path is designed in verilog using the Vivado design suite.

After the final design is synthesized, the processor can be simulated to view the inputs and outputs. A bitstream which describes the RTL schematic can be generated from the Vivado project, to test the design on an FPGA board.

The FPGA board can be programmed using the bitstream generated from the Vivado design suite. Inputs can be given through push buttons and slide switches and outputs can be represented by LEDs already provided on the board.

#### 1. Vivado Design Suite - HLx Editions

Download the ‘Full Product Installation’ from the Xilinx website and install it in your computer.

There are Windows and Linux versions available.

During installation, perform the standard, default installation.

<https://www.xilinx.com/support/download.html>

#### 2. Basys 3 FPGA Board

The board will be available during office hours, if you want to test out your design. This testing will be done by the TAs.

Refer to the User Manual for details on how to configure the Vivado Project for this particular board.

[https://reference.digilentinc.com/\\_media/reference/programmable-logic/basys-3/basys3\\_rm.pdf](https://reference.digilentinc.com/_media/reference/programmable-logic/basys-3/basys3_rm.pdf)

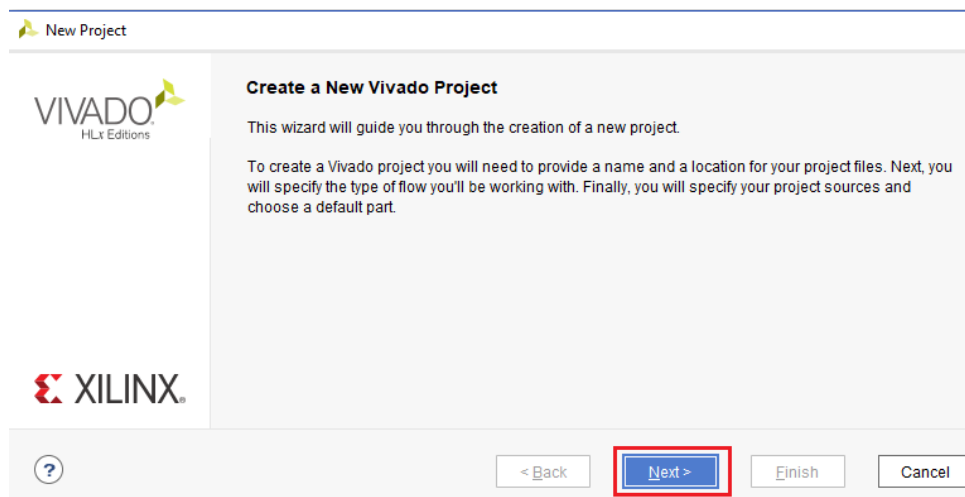
## APPENDIX

### Setup:

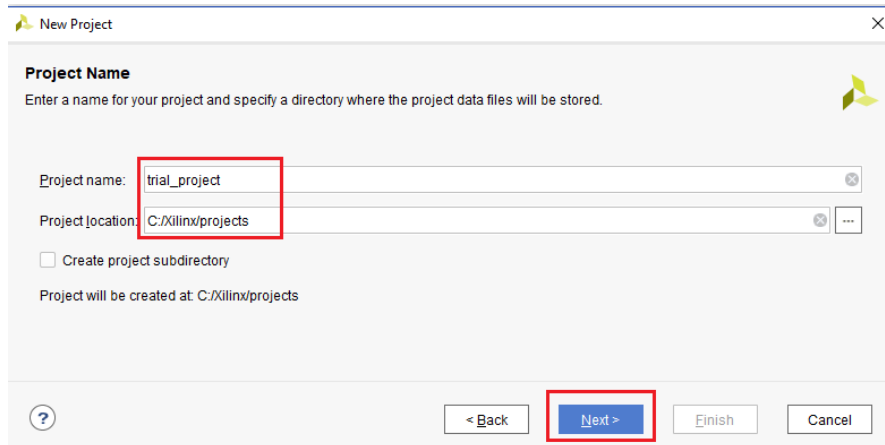
1. After start-up, click on the 'Create Project' button and then hit 'Next'.



- 2.



3. Name your project and specify a particular path. Make sure the project path location does not have spaces.



**New Project**

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

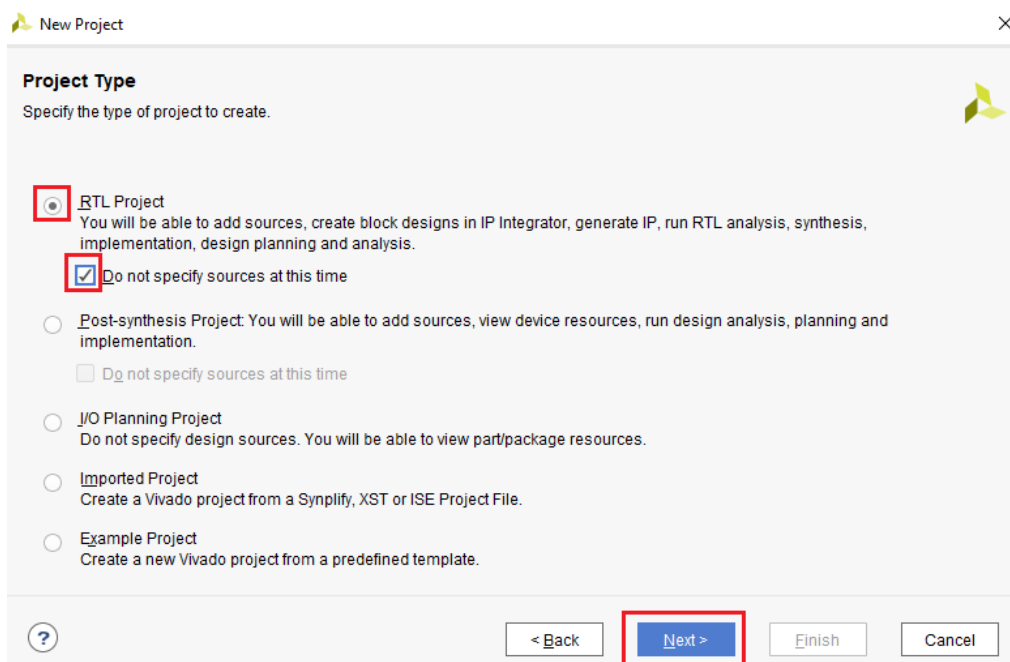
Project name:

Project location:

☐ Create project subdirectory

Project will be created at: C:/Xilinx/projects

4. Choose 'RTL Project' and check the box that says 'Do not specify sources at this time'.



**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**

☐ **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, XST or ISE Project File.

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

5. The Basys 3 FPGA board uses an Artix 7 FPGA. From the datasheet and user manual, select all the specific options as shown below.

New Project X

### Default Part

Choose a default Xilinx part or board for your project. This can be changed later.

**Parts** | Boards

[Reset All Filters](#)

Category:  Package:  Temperature:

Family:  Speed:



Search:

Part	I/O Pin Count	Available IOBs	LUT Elements	FlipFlops	Block RAMs	Ultra RAMs	DSPs
xc7a15tcpg236-1	236	106	10400	20800	25	0	45
xc7a35tcpg236-1	236	106	20800	41600	50	0	90
xc7a50tcpg236-1	236	106	32600	65200	75	0	120

?

6.

New Project X

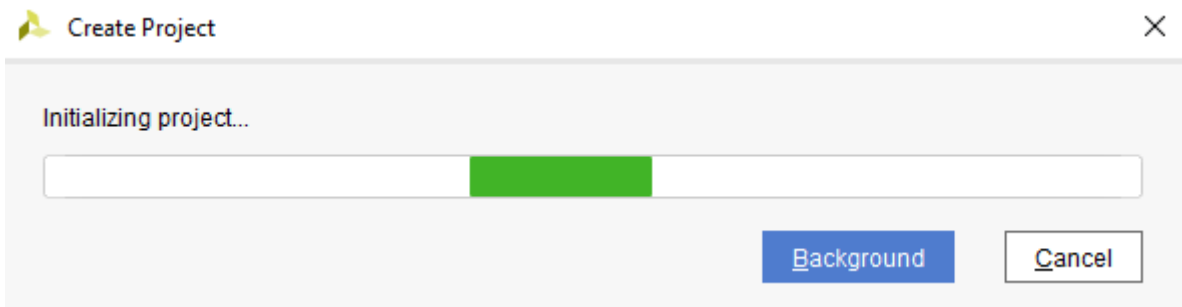
### New Project Summary

- i A new RTL project named 'trial\_project' will be created.
- i The default part and product family for the new project:  
Default Part: xc7a35tcpg236-1  
Product: Artix-7  
Family: Artix-7  
Package: cpg236  
Speed Grade: -1

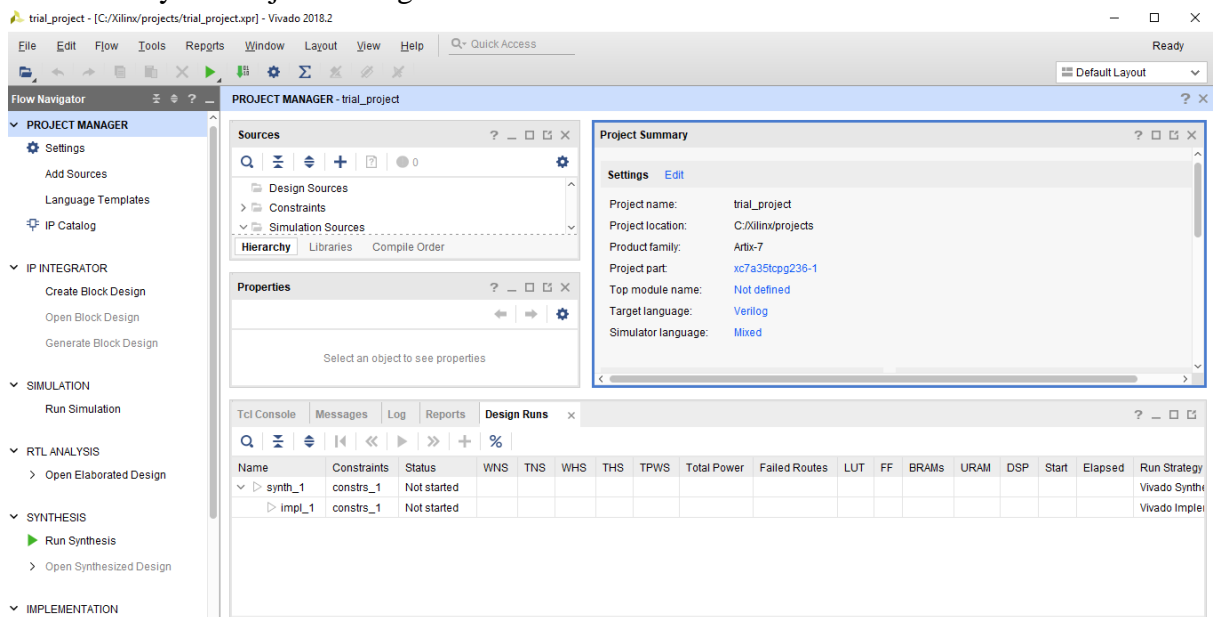
To create the project, click Finish

?

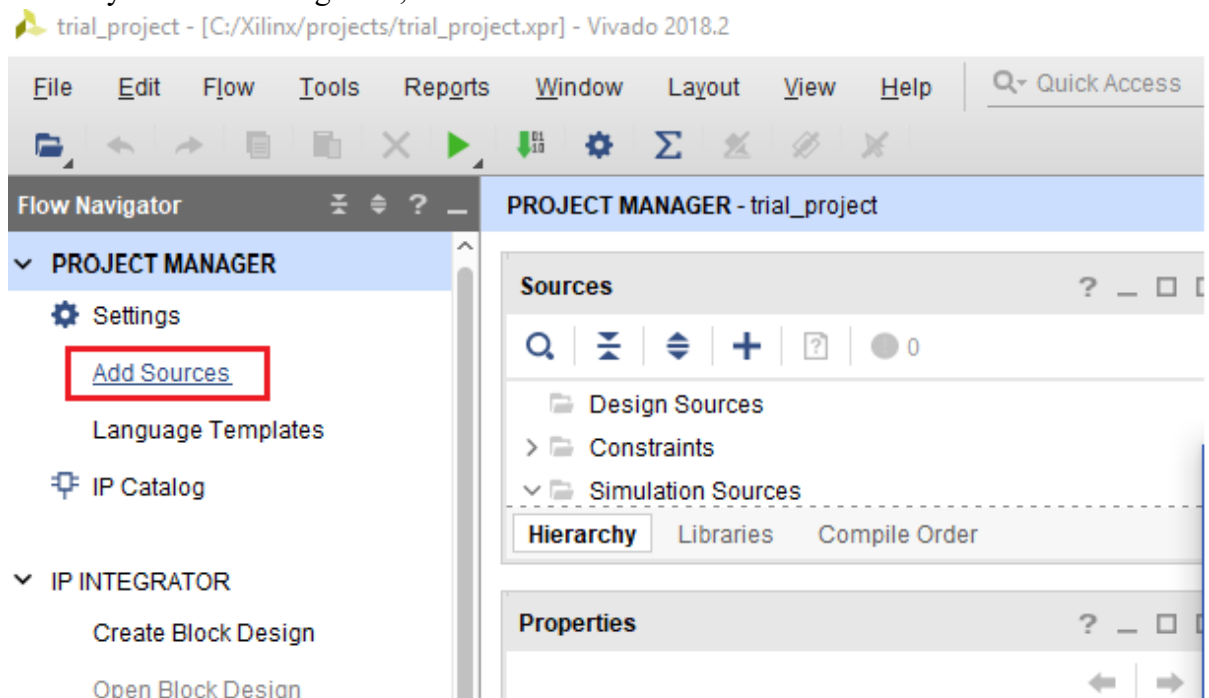
7. This might take a few seconds.



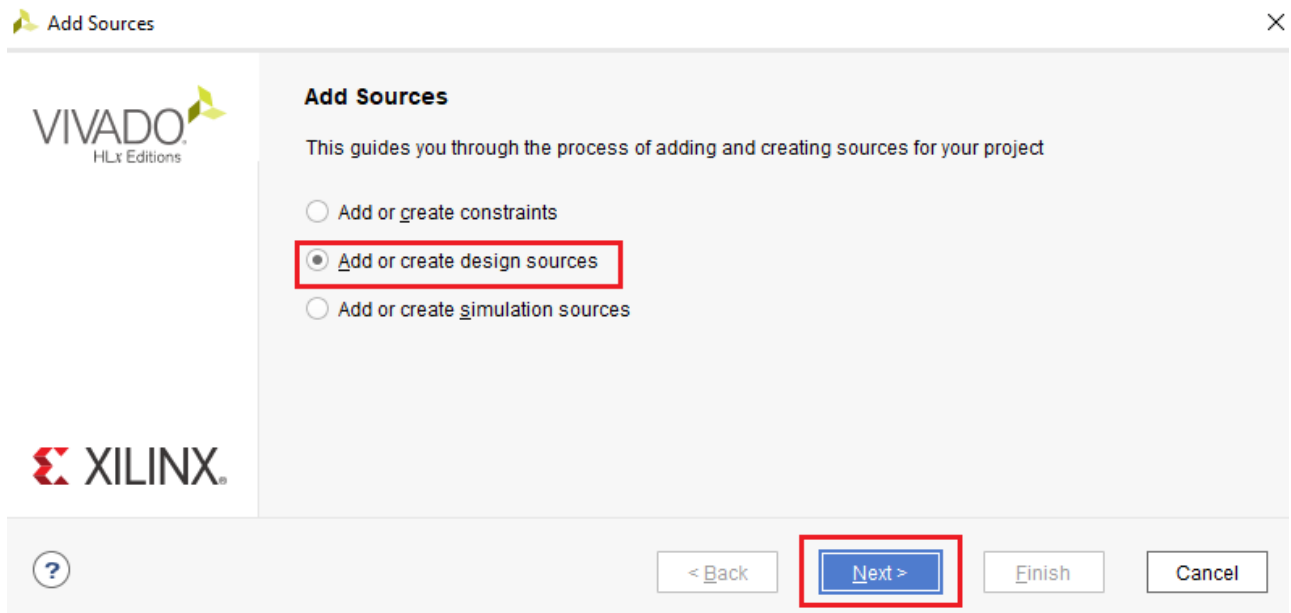
8. Make sure your Project Manager looks like this.



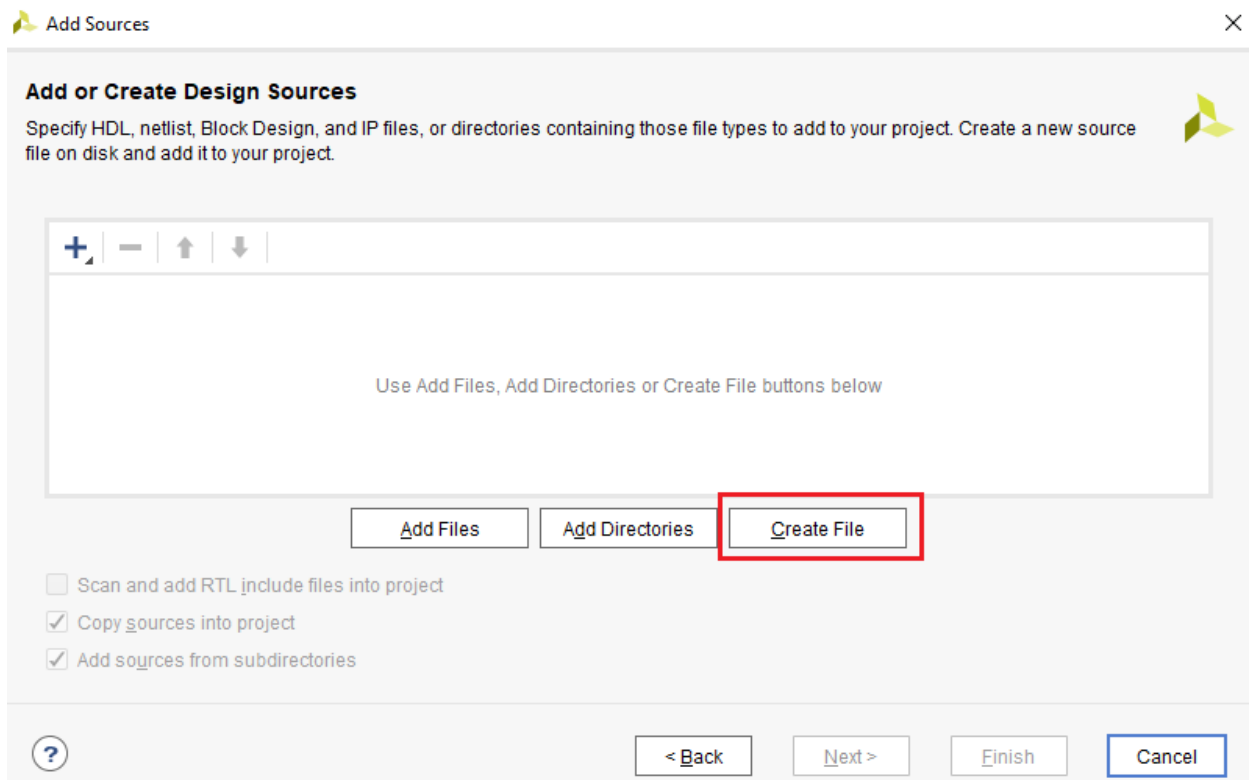
9. To add your first Verilog code, click 'Add Sources'



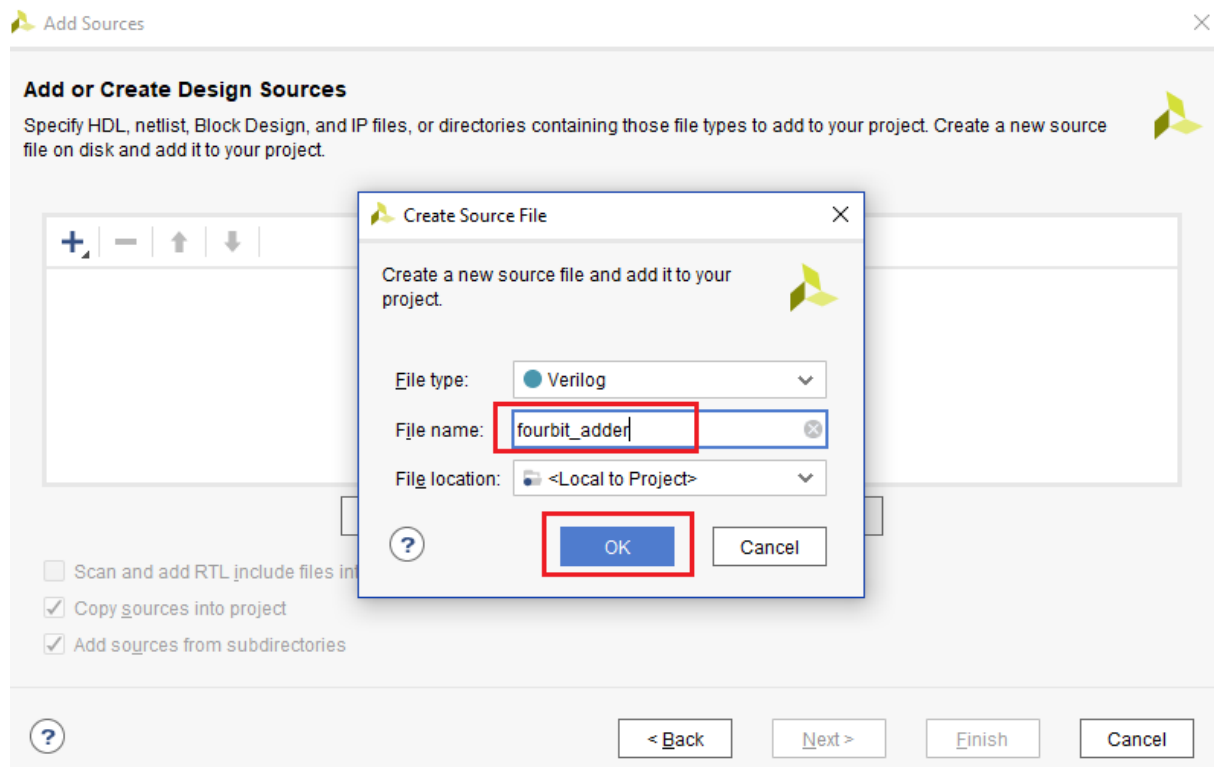
10. Click on 'Add or create design sources'.



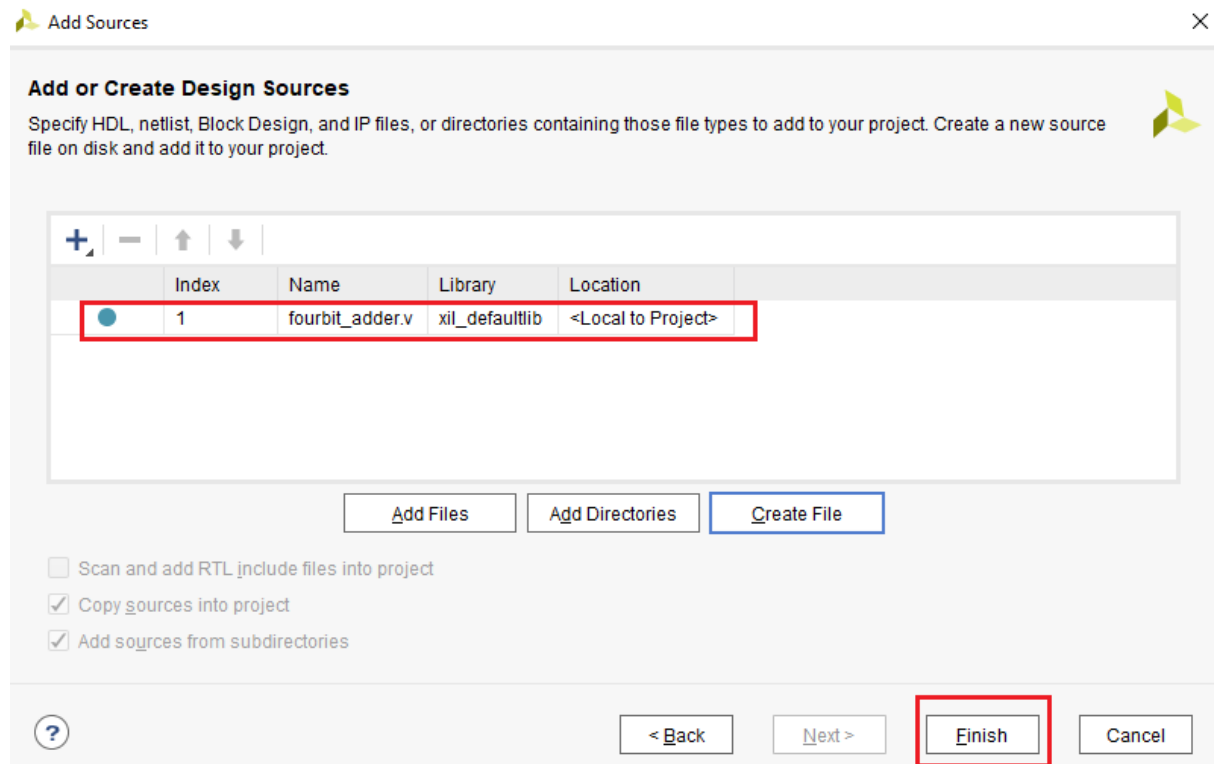
11. If your verilog code (.v) has already been created, click 'Add files' and browse to select the corresponding file. If you need to create a new verilog code, click 'Create file'.



12. Name the file and hit 'OK'




13. This is an example for a 4-bit adder, so there is only one verilog file created here. In case of the 8-bit controller, a verilog file has to be created for each component. Click 'Finish'. You can add other sources later as well.



14. Define the input and output ports and bus width if the ports are more than a bit wide. This step is optional. You can hit 'Cancel' and decide on how many ports you need as you write the code.



 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**

Module name:

**I/O Port Definitions**

Port Name	Direction	Bus	MSB	LSB
a	input	<input checked="" type="checkbox"/>	3	0
b	input	<input checked="" type="checkbox"/>	3	0
sum	output	<input checked="" type="checkbox"/>	3	0
carry	output	<input type="checkbox"/>	0	0

? OK Cancel

15. This is how your project manager should look like.

PROJECT MANAGER - trial\_project

**Sources**

Design Sources (1)

- fourbit\_adder (fourbit\_adder.v)

Constraints

Hierarchy Libraries Compile Order

**Source File Properties**

fourbit\_adder.v

General Properties

Project Summary

fourbit\_adder.v

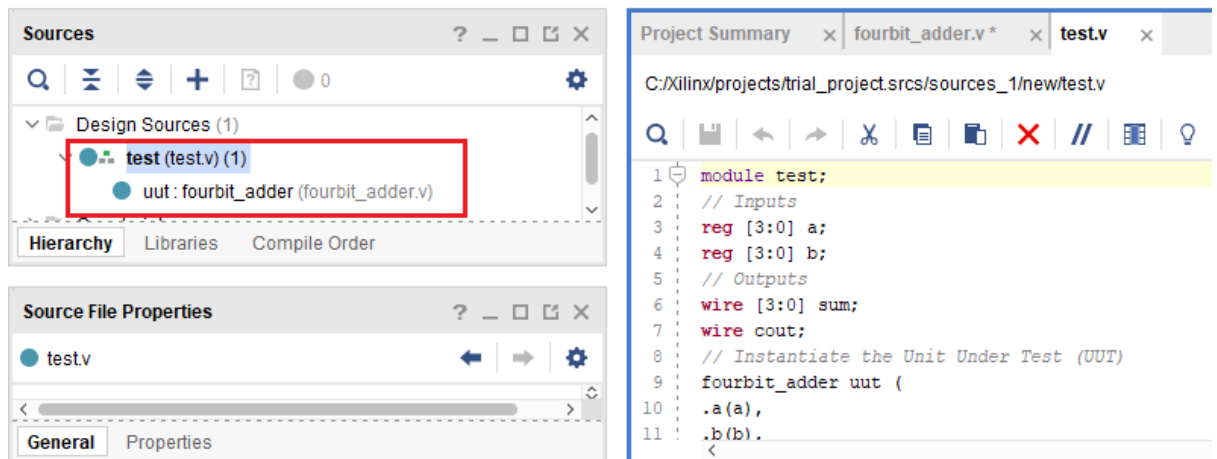
C:/Xilinx/projects/trial\_project/srcs/sources\_1/new/fourbit\_adder.v

```

21
22
23 module fourbit_adder(
24     input [3:0] a,
25     input [3:0] b,
26     output [3:0] sum,
27     output carry
28 );
29 endmodule
30

```

16. Repeat Steps 9-14 to create a test bench verilog code that tests the design in the previous code with various test cases.



17. The test bench can then be simulated by selecting 'Run Simulation' and then 'Run behavioral simulation' in the Project manager.