CSE 490/590 COMPUTER ARCHITECTURE – SUMMER 2019

PROJECT 1 – DESIGN OF AN 8-BIT NON-PIPELINED PROCESSOR USING VERILOG

DUE : 26 June 2019, 8:30am

PART 1

This is a group project. Each group can have 2 students. A final project report is due on 26 June 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, 5th 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. add
ii. addi
iii. sub
iv. lw
v. sw
vi. jmp

Instruction	Opcode
add	000
addi	001
sub	010
lw	011
sw	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

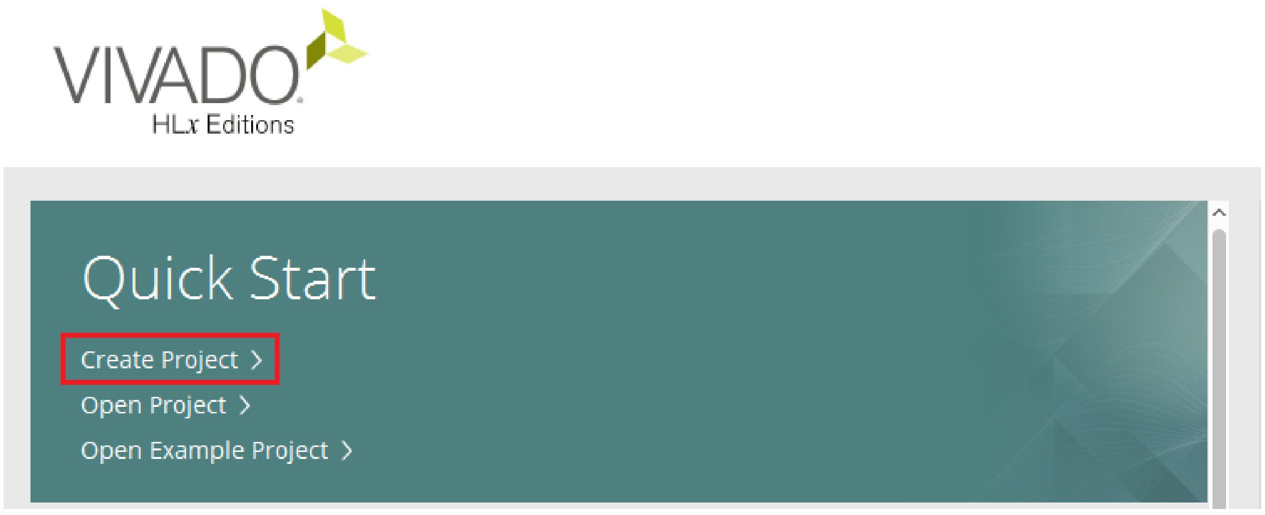
2. **Basys 3 FPGA Board**

   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

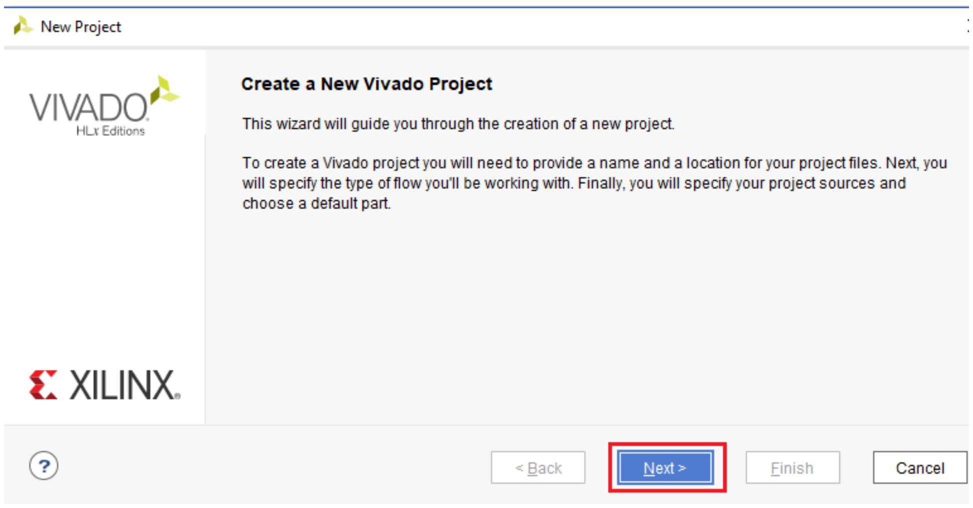
APPENDIX

Setup:

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



2.

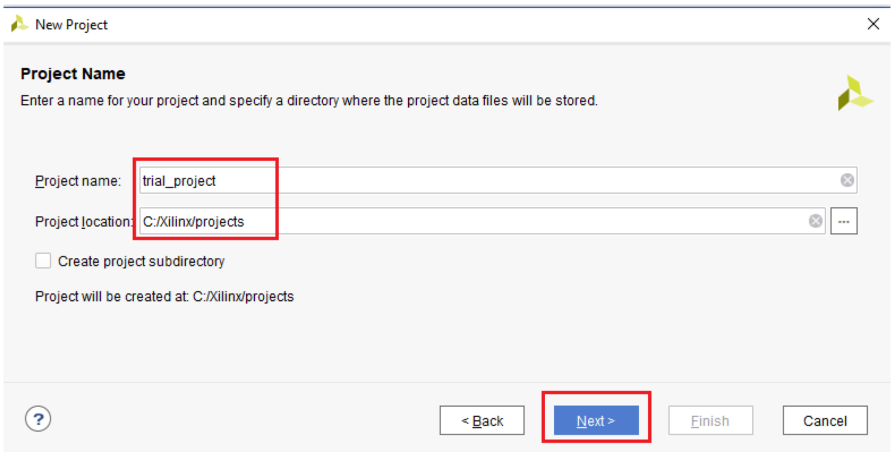


Create a New Vivado Project

This wizard will guide you through the creation of a new project.

To create a Vivado project you will need to provide a name and a location for your project files. Next, you will specify the type of flow you'll be working with. Finally, you will specify your project sources and choose a default part.

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



Project Name

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

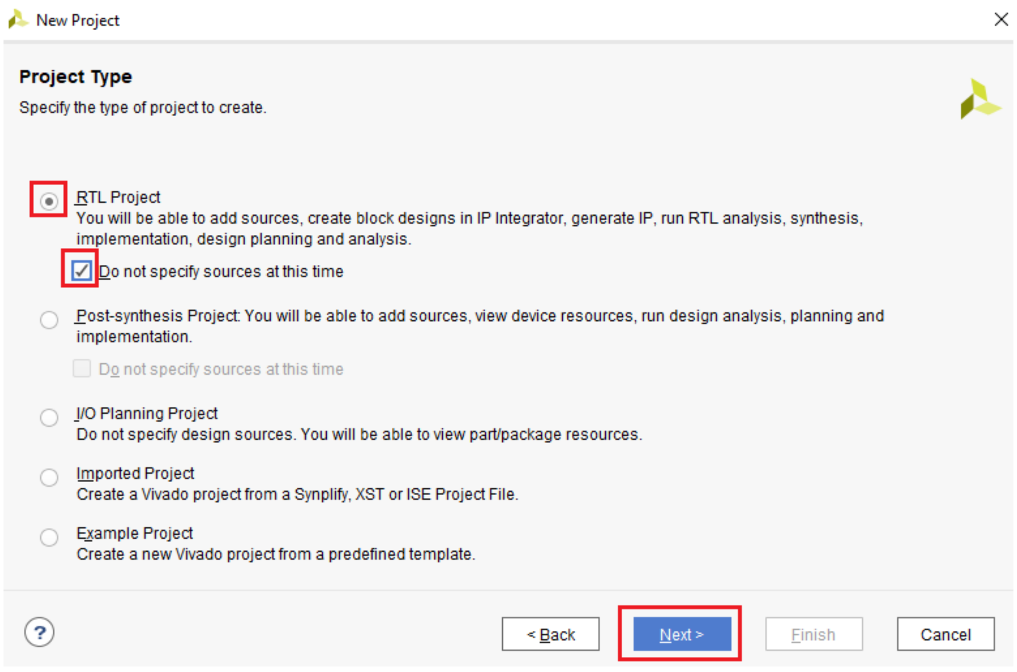
Project name: trial_project

Project location: C:/Xilinx/projects

☐ 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'.



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.

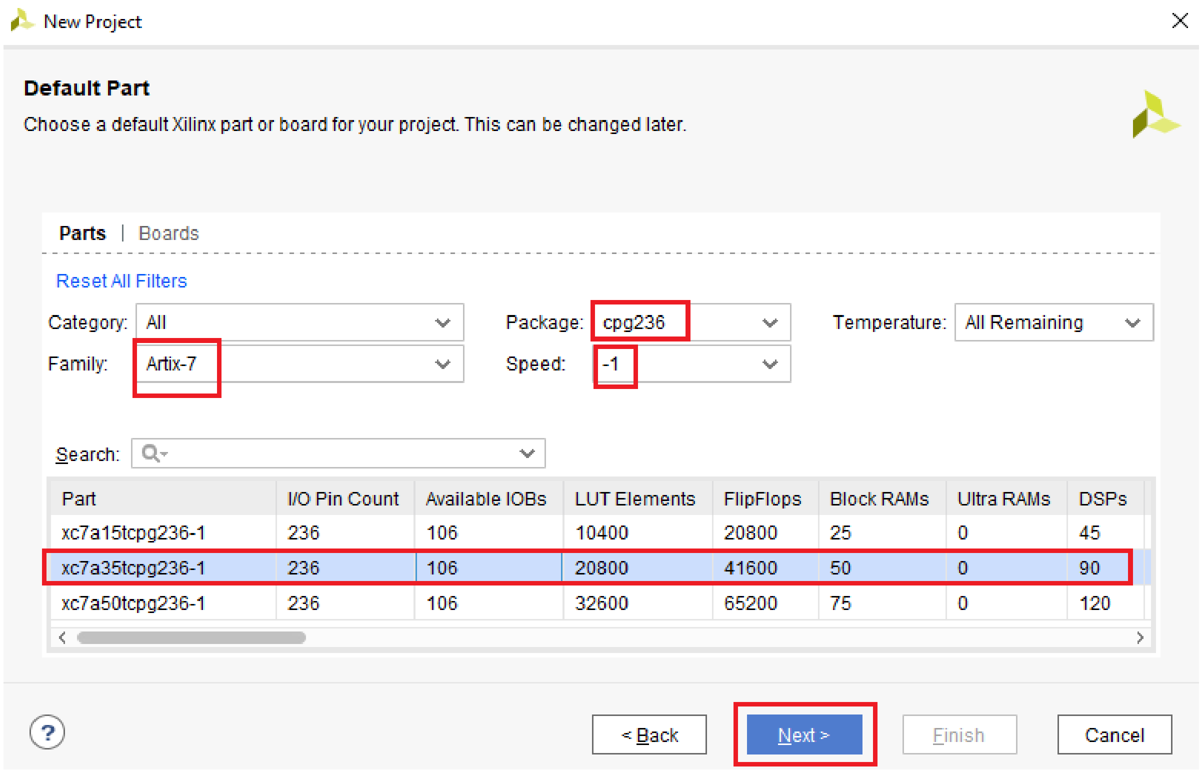
Do not specify sources at this time

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.



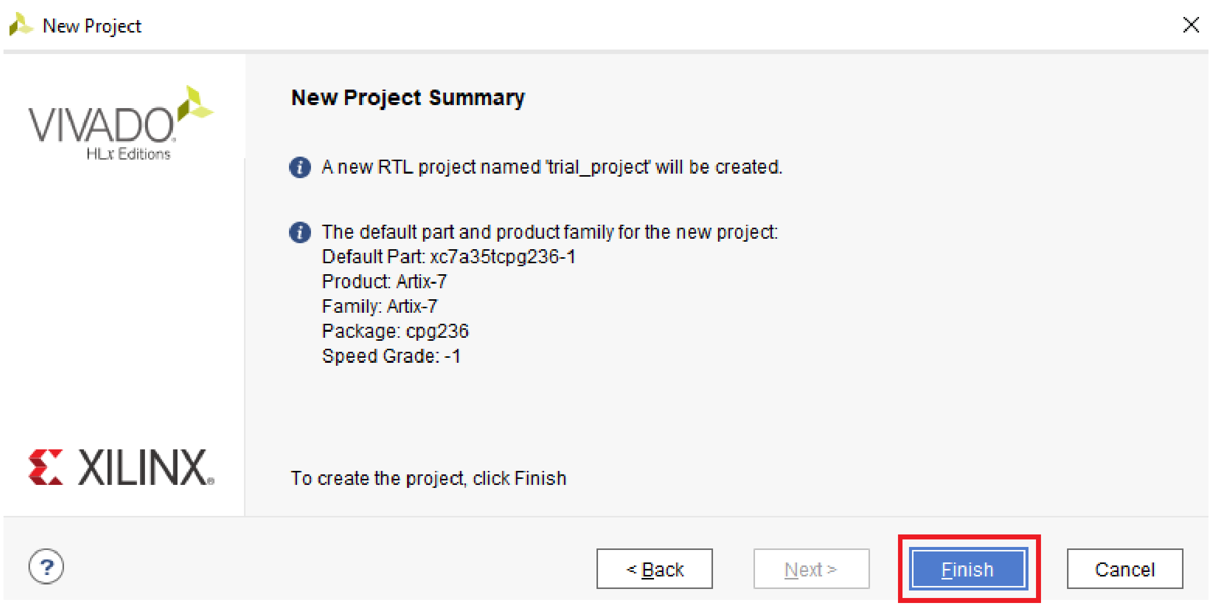
Default Part

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

Category: All
Family: Artix-7
Package: cpg236
Speed: -1
Temperature: All Remaining
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.



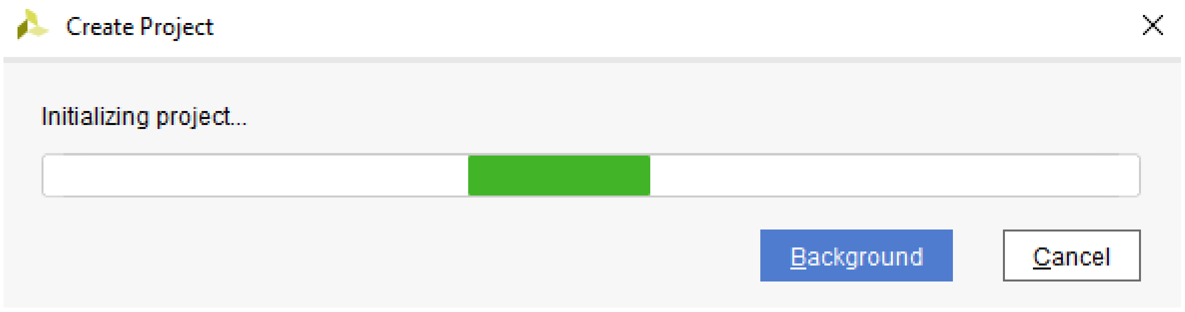
New Project Summary

A new RTL project named 'trial_project' will be created.

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

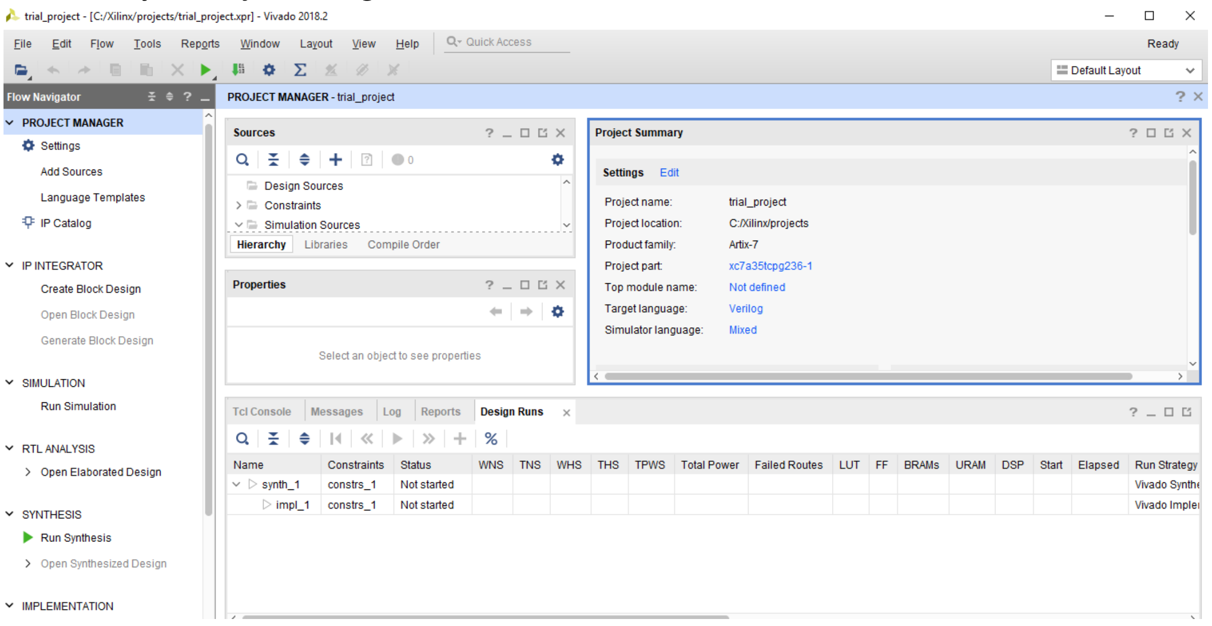
To create the project, click Finish

7. This might take a few seconds.



Initializing project...

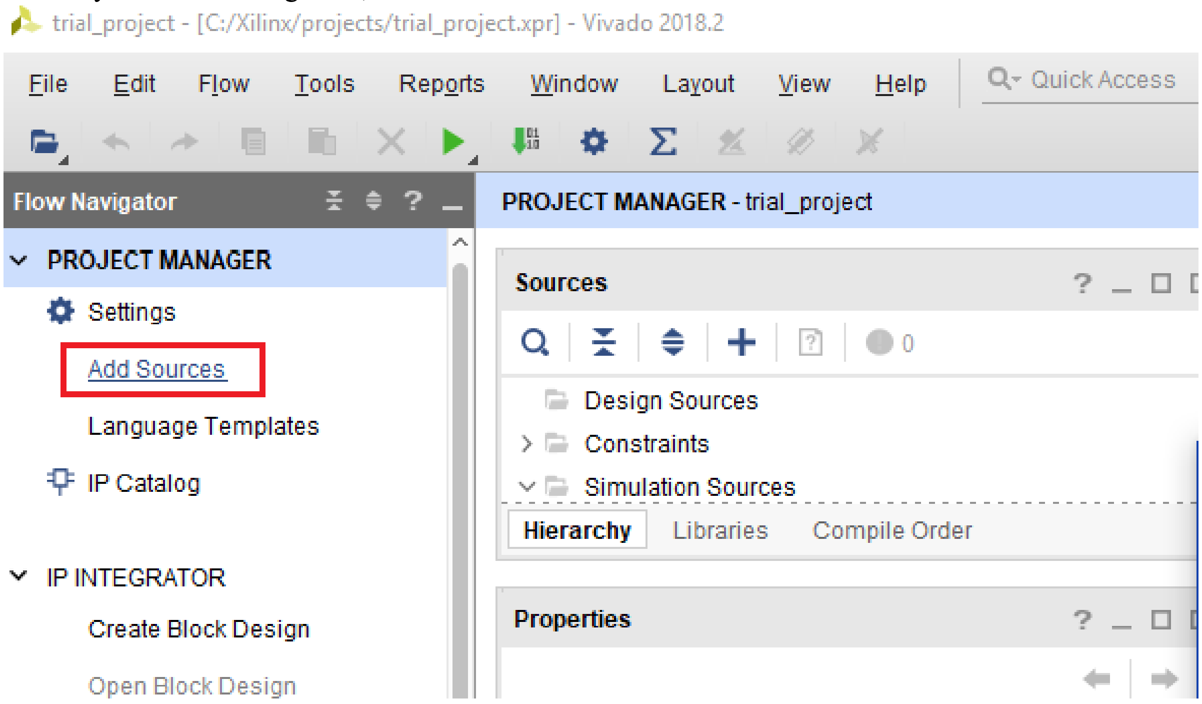
8. Make sure your Project Manager looks like this.



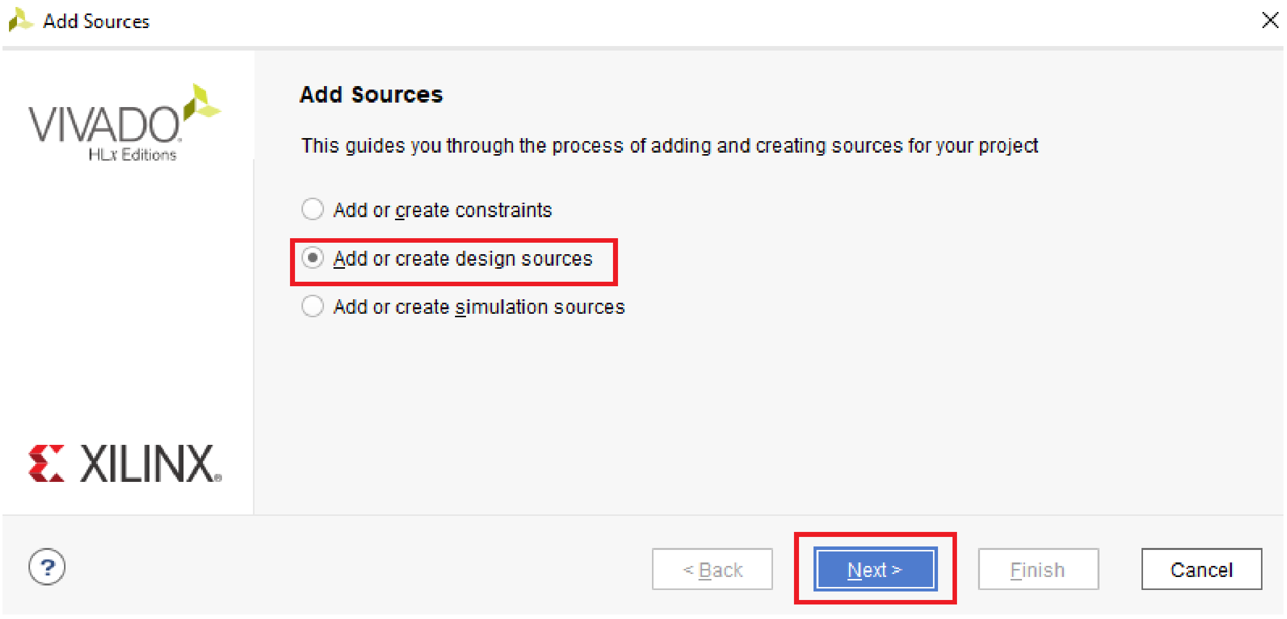
Project name:	trial_project
Project location:	C:/Xilinx/projects
Product family:	Artix-7
Project part:	xc7a35tcpg236-1
Top module name:	Not defined
Target language:	Verilog
Simulator language:	Mixed

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP	Start	Elapsed	Run Strategy
synth_1	constrs_1	Not started															Vivado Synthe
impl_1	constrs_1	Not started															Vivado Imple

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



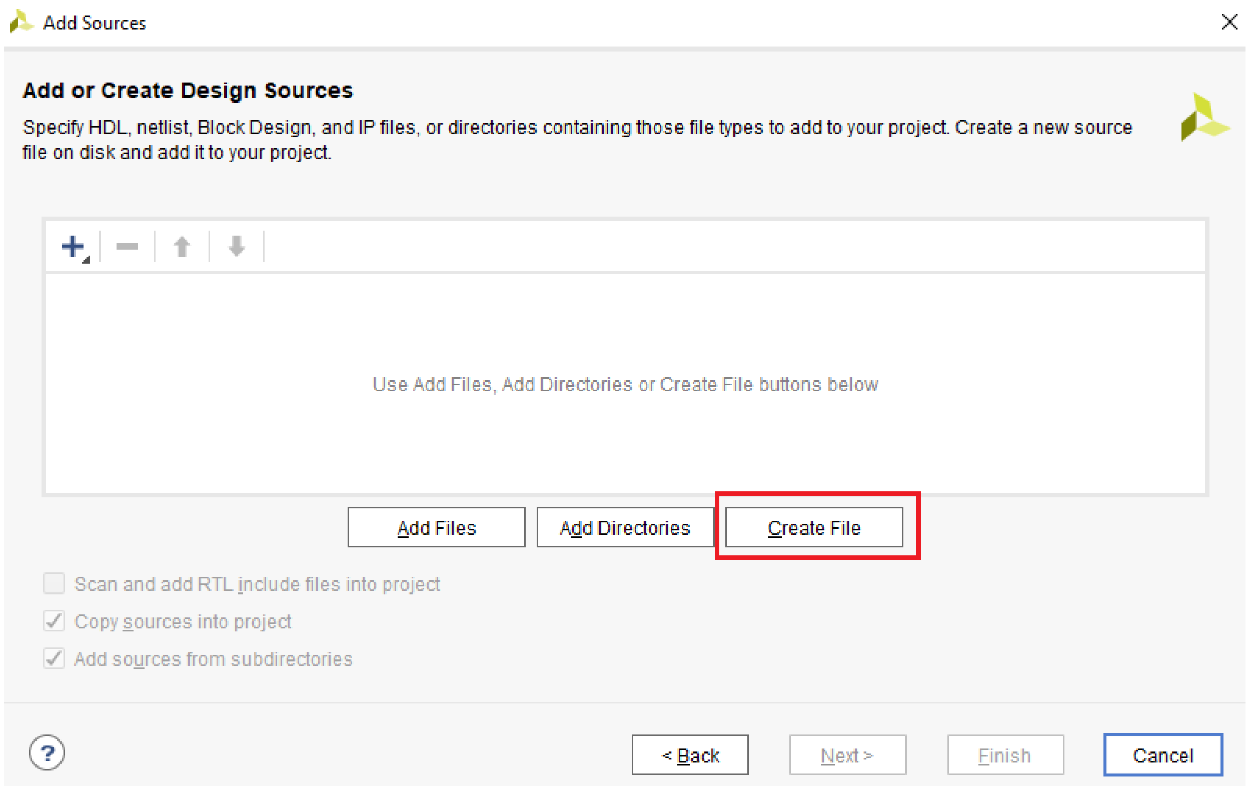
10. Click on 'Add or create design sources'.


Add Sources
VIVADO HLx Editions
XILINX
Add or create constraints
Add or create design sources
Add or create simulation sources
< Back
Next >
Finish
Cancel

Add Sources

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

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'.



Add Sources

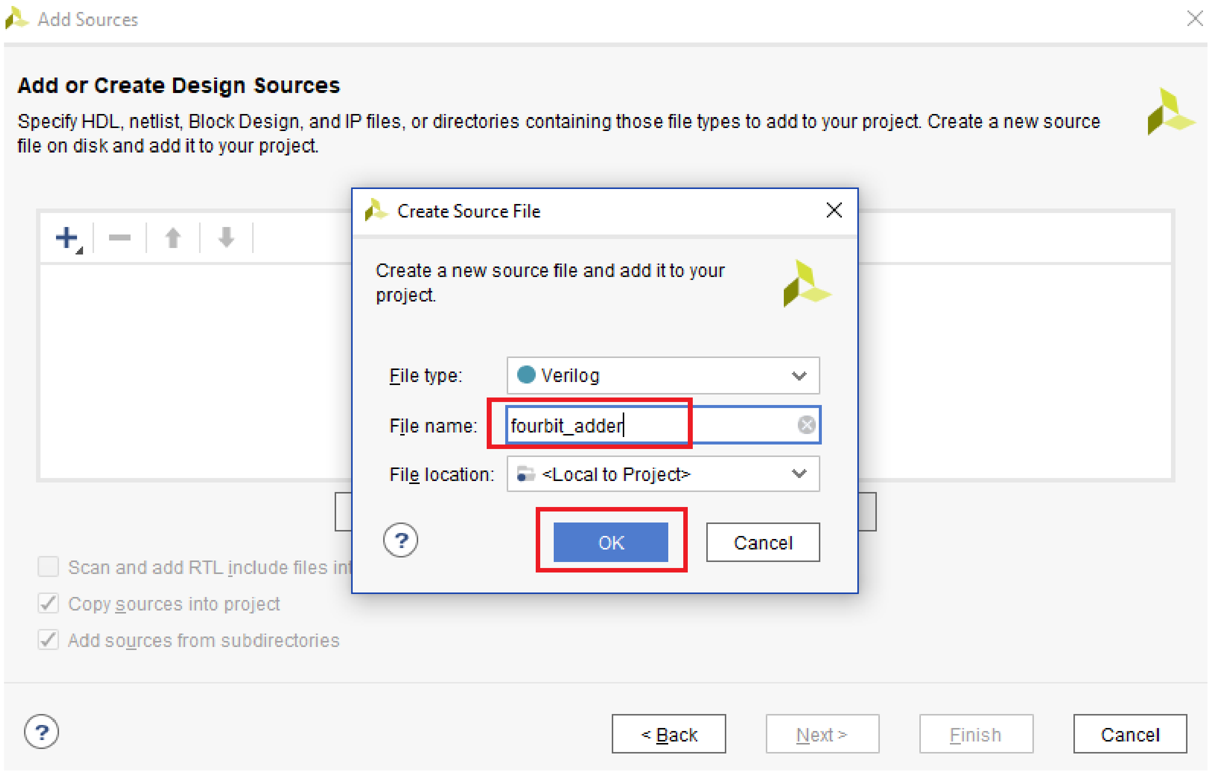
Add or Create Design Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.

Use Add Files, Add Directories or Create File buttons below

Add Files
Add Directories
Create File
Scan and add RTL include files into project
Copy sources into project
Add sources from subdirectories
< Back
Next >
Finish
Cancel

12. Name the file and hit ‘OK’



Add Sources

Add or Create Design Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.

Create Source File

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

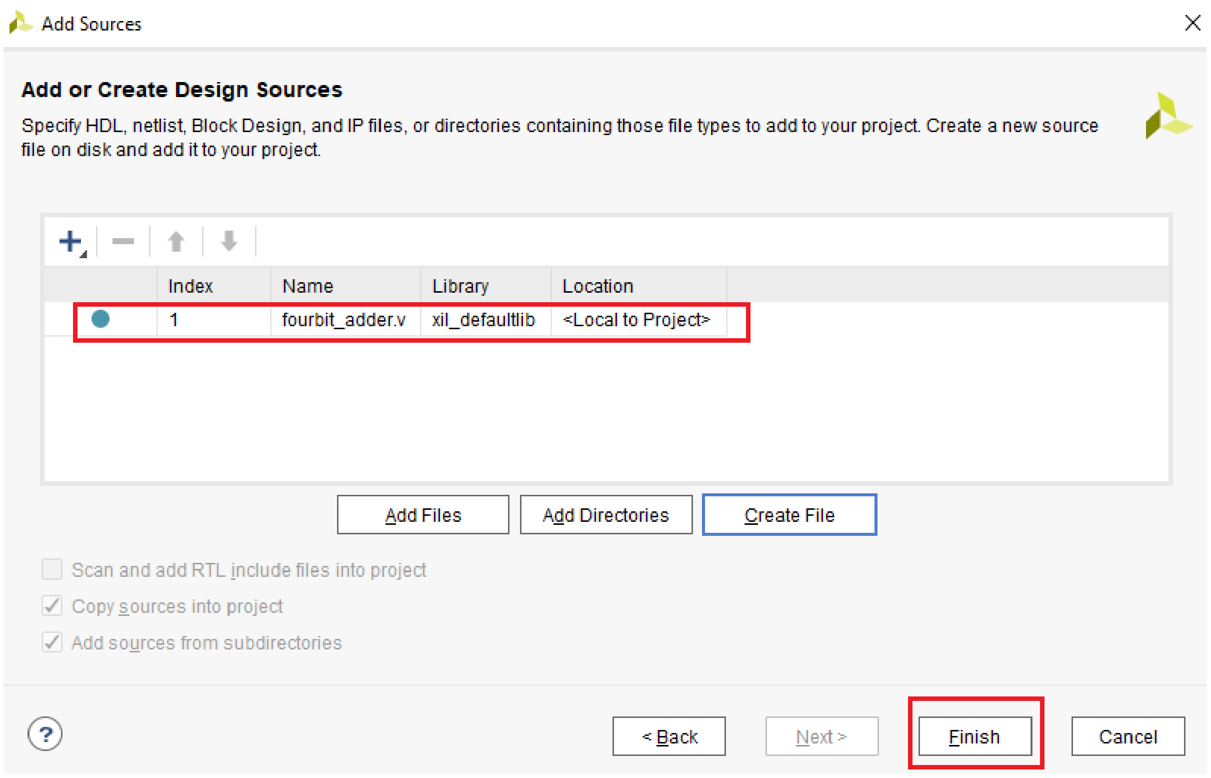
File type: Verilog

File name: fourbit_adder

File location: <Local to Project>

OK
Cancel
Scan and add RTL include files int
Copy sources into project
Add sources from subdirectories
< Back
Next >
Finish
Cancel

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.



Add Sources

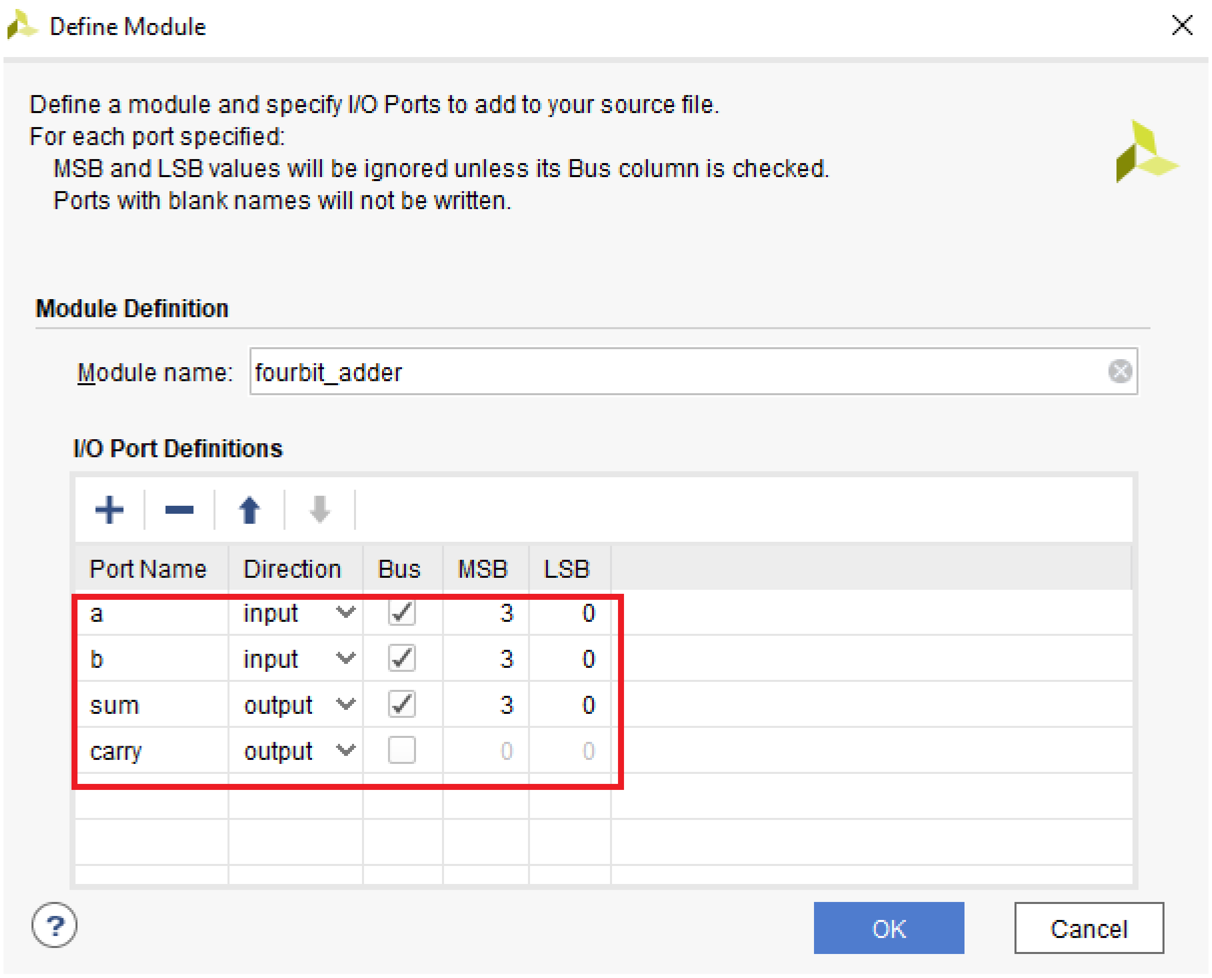
Add or Create Design Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.

	Index	Name	Library	Location
●	1	fourbit_adder.v	xil_defaultlib	<Local to Project>

Add Files
Add Directories
Create File
Scan and add RTL include files into project
Copy sources into project
Add sources from subdirectories
< Back
Next >
Finish
Cancel

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.

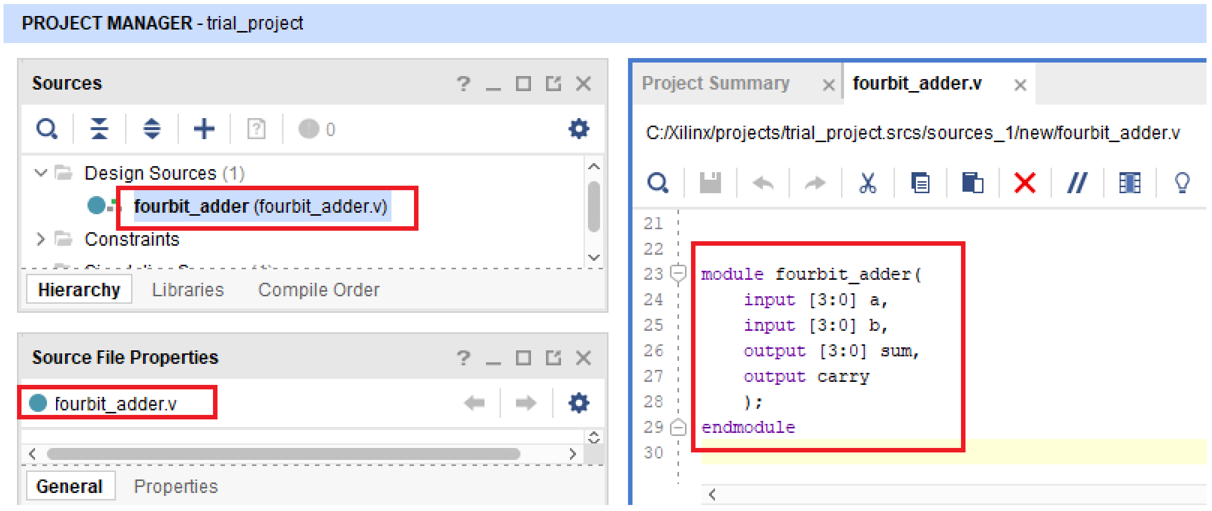
Module Definition

Module name: fourbit_adder

I/O Port Definitions

Port Name	Direction	Bus	MSB	LSB
a	input	✓	3	0
b	input	✓	3	0
sum	output	✓	3	0
carry	output		0	0

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



PROJECT MANAGER - trial_project

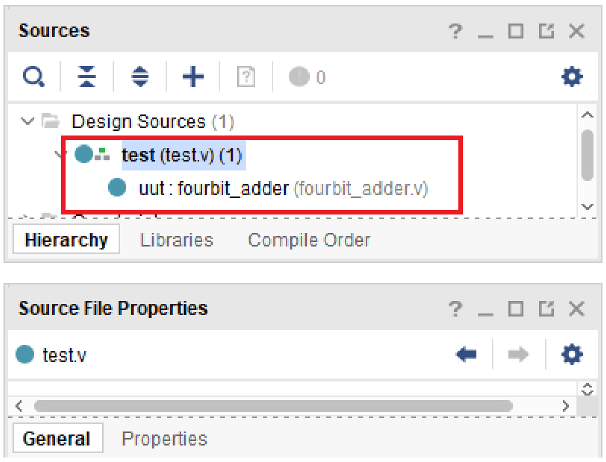
Sources

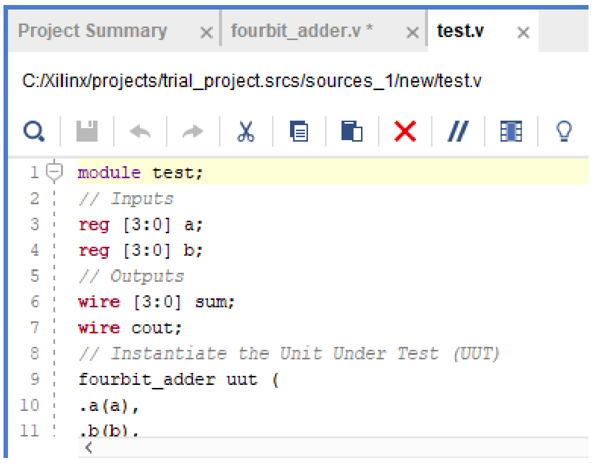
Design Sources (1)
fourbit_adder (fourbit_adder.v)
Constraints

C:/Xilinx/projects/trial_project.srcs/sources_1/new/fourbit_adder.v

```
module fourbit_adder(
    input [3:0] a,
    input [3:0] b,
    output [3:0] sum,
    output carry
    );
endmodule
```

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





```
module test;
// Inputs
reg [3:0] a;
reg [3:0] b;
// Outputs
wire [3:0] sum;
wire cout;
// Instantiate the Unit Under Test (UUT)
fourbit_adder uut (
.a(a),
.b(b),
```

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