Laboratory 2: Analysis and Synthesis of Combinational Circuits

Learning Objectives

- Learn the basics of Verilog
- Synthesize simple combinational circuits
- Analyze simple combinational cirucits
- Use of minimization approaches
- Timing hazards

In this lab, you will apply circuit minimization techniques to design and assemble simple combinational circuits. You also learn to work with Xilinx Vivado to synthesize and simulate circuits.

The theoretical background required for this lab are

- Boolean algebra
- Karnaugh maps
- Minimization approaches
- Detecting and eliminating timing hazards

1. Pre-Lab

For the prelab, you are asked to syntehsize, simulate and analyze logic functions. Additionally you are asked to use Vivado and present the Register Transfer Logic (RTL) diagram and the timing diagram with all the possible combinations for the inputs.

1.1 Synthesis of A Simple Combinational Circuit

Consider the following logic function

$$\mathrm{F}(\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}) = \sum(1,3,5,6,7,14).$$

1.1.1. Circuit Minimization

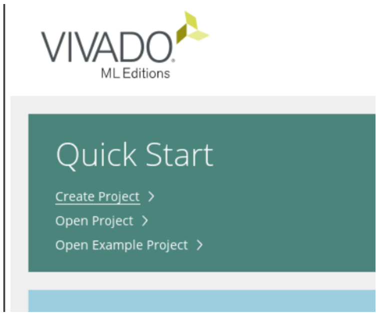
1. Derive the Karnaugh map for function $\mathrm{F}(\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D})$. Include this in your prelab report.
2. Find the minimal sum-of-product for $\mathrm{F}(\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D})$. Include this in your prelab report.
3. Draw the NAND-NAND implementation of this function. Include the diagram in your prelab report.
4. Complete the truth table for function $\mathrm{F}(\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D})$. Include the completed table in your prelab report.

1.1.2 Verilog and Simulation

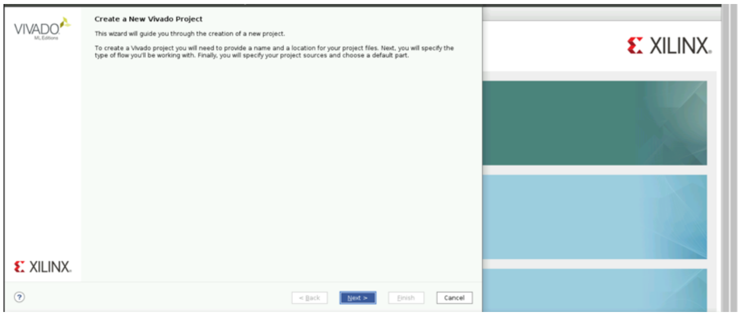
1. Follow the instructions described "**Vivado-How to Run.pdf**" to run Vivado.

2. Create a Verilog file (*.v). Use the following instructions to create and synthesize your circuit for implementing function $F(A, B, C, D)$.

   a. Click on "Create Project"



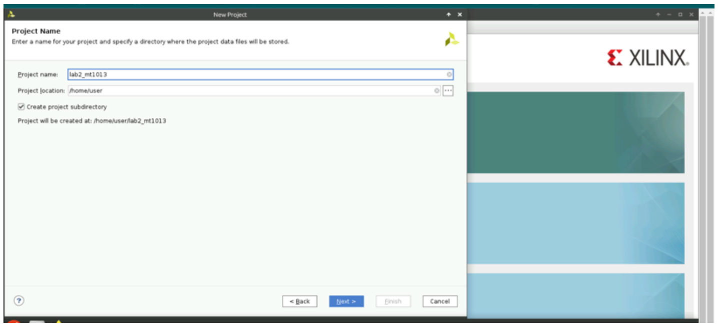
   b. Click "Next"


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.

   c. Enter the name of the project and click "Next"


New Project

Project Name

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

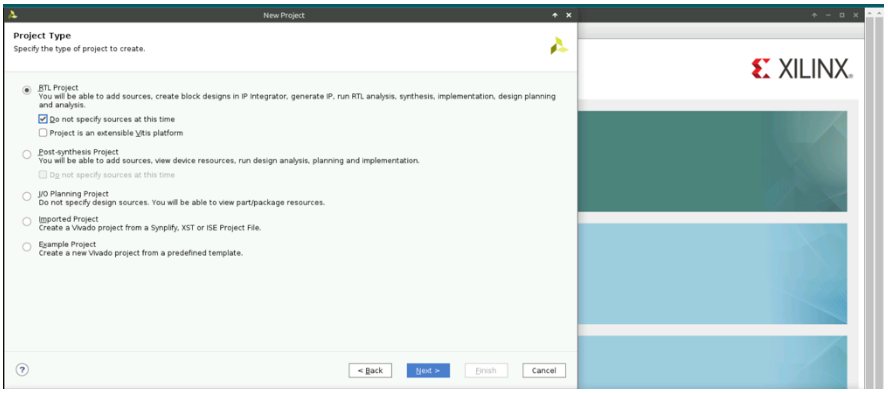
Project name: lab2_mt1013

Project location: /home/user

☑ Create project subdirectory

Project will be created at: /home/user/lab2_mt1013

d. Choose RTL Project and check “Do Not Specify Resources at this time”. Click “Next”.



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

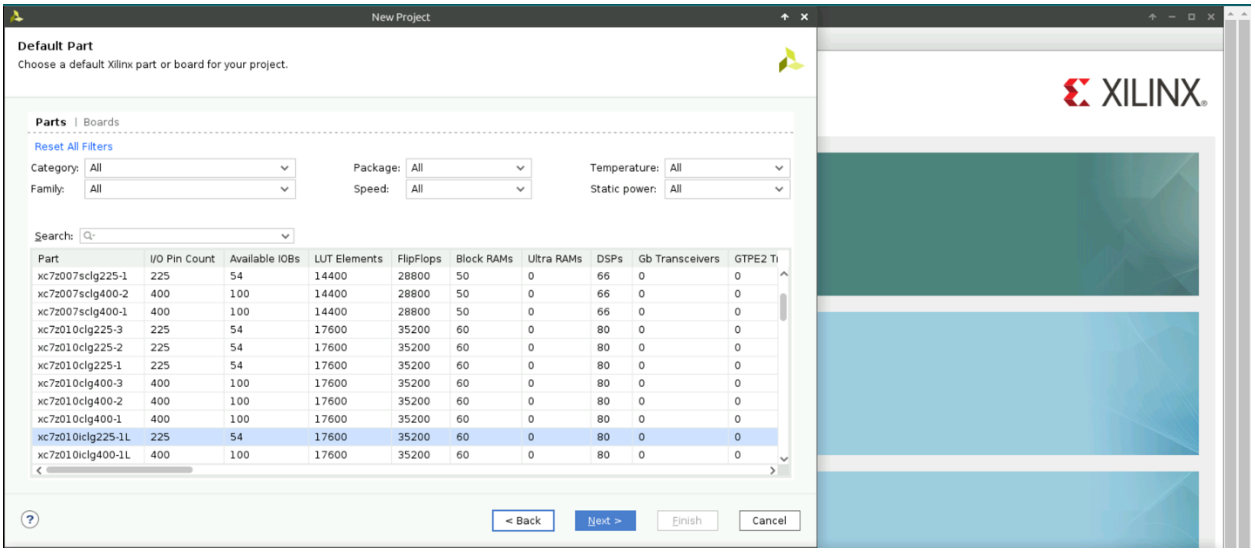
○ Post-synthesis Project
You will be able to add sources, view device resources, run design analysis, planning and implementation.
□ Do not specify sources at this time

○ 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, XST or ISE Project File.

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

e. Choose the default part as shown in the screenshot below. You do not need to make any changes on this page. Click “Next”



Default Part

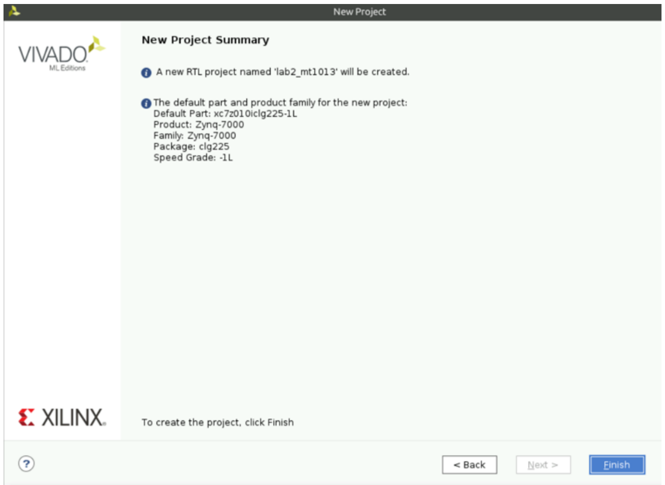
Choose a default Xilinx part or board for your project.

Category: All
Family: All
Package: All
Speed: All
Temperature: All
Static power: All

Search: Q-

Part	I/O Pin Count	Available IOBs	LUT Elements	FlipFlops	Block RAMs	Ultra RAMs	DSPs	Gb Transceivers	GTPE2 Tı
xc7z007sclg225-1	225	54	14400	28800	50	0	66	0	0
xc7z007sclg400-2	400	100	14400	28800	50	0	66	0	0
xc7z007sclg400-1	400	100	14400	28800	50	0	66	0	0
xc7z010clg225-3	225	54	17600	35200	60	0	80	0	0
xc7z010clg225-2	225	54	17600	35200	60	0	80	0	0
xc7z010clg225-1	225	54	17600	35200	60	0	80	0	0
xc7z010clg400-3	400	100	17600	35200	60	0	80	0	0
xc7z010clg400-2	400	100	17600	35200	60	0	80	0	0
xc7z010clg400-1	400	100	17600	35200	60	0	80	0	0
xc7z010iclg225-1L	225	54	17600	35200	60	0	80	0	0
xc7z010iclg400-1L	400	100	17600	35200	60	0	80	0	0

f. Click “Finish”. This will create your project in Vivado.



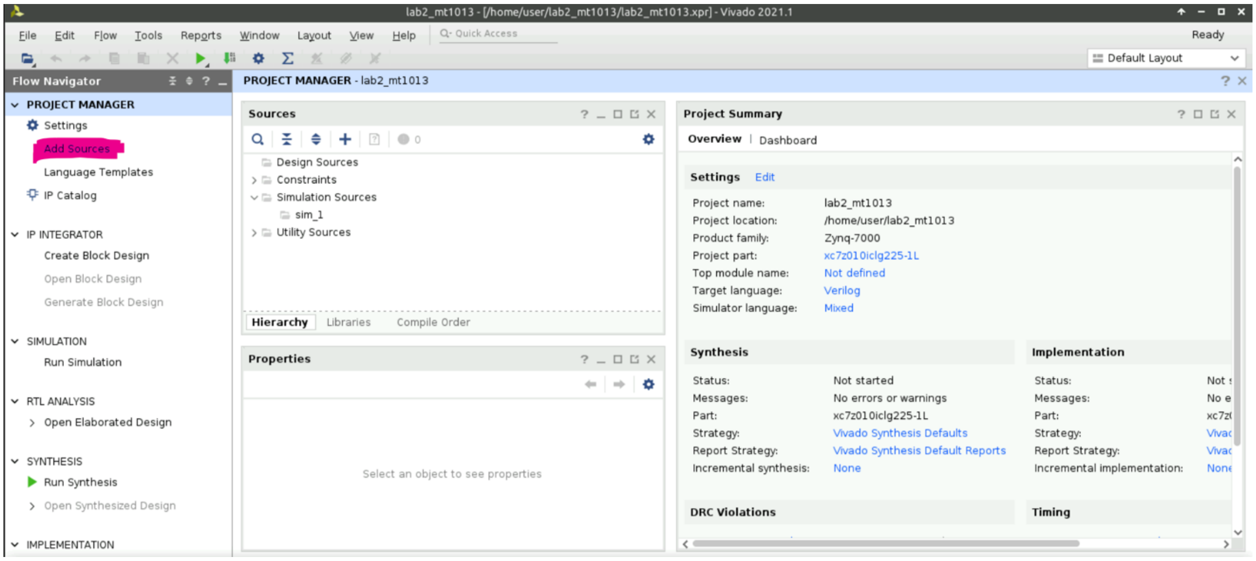
New Project Summary

A new RTL project named 'lab2_mt1013' will be created.

The default part and product family for the new project:
Default Part: xc7z010iclg225-1L
Product: Zynq-7000
Family: Zynq-7000
Package: clg225
Speed Grade: -1L

To create the project, click Finish

g. You will see the following screen after creating your project. Click on “Add Sources”. This is highlighted in the following screenshot.



Settings Edit

Project name: lab2_mt1013
Project location: /home/user/lab2_mt1013
Product family: Zynq-7000
Project part: xc7z010iclg225-1L
Top module name: Not defined
Target language: Verilog
Simulator language: Mixed

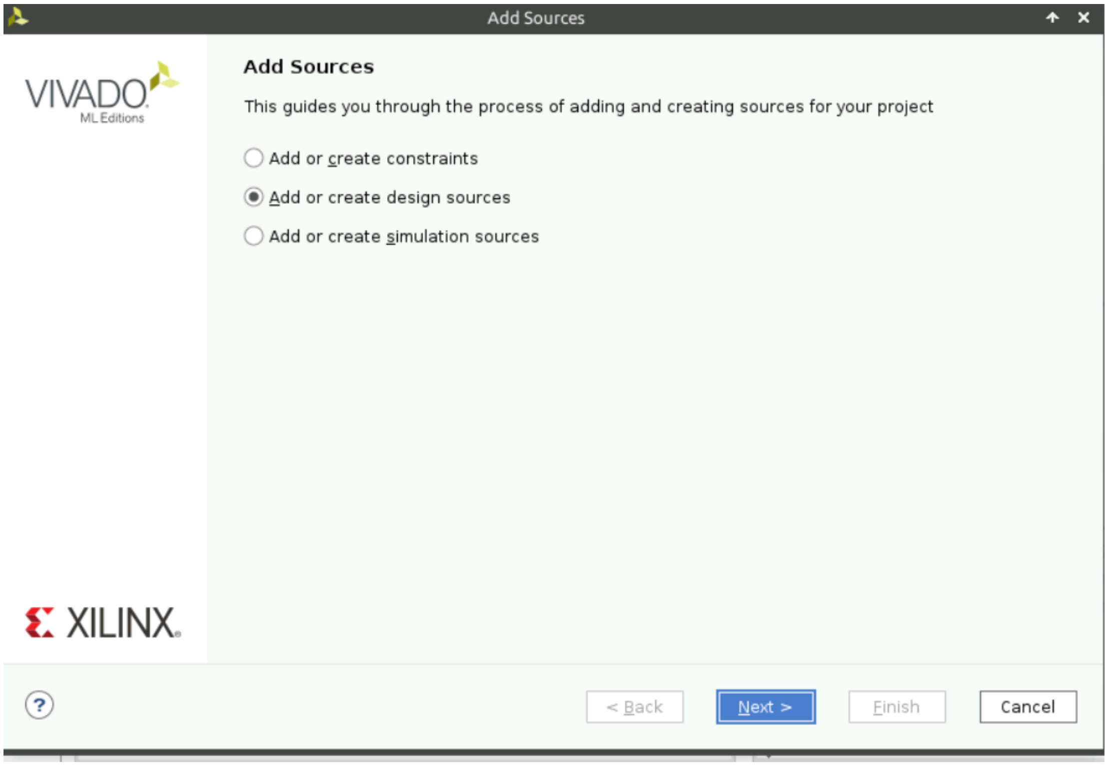
Synthesis

Status: Not started
Messages: No errors or warnings
Part: xc7z010iclg225-1L
Strategy: Vivado Synthesis Defaults
Report Strategy: Vivado Synthesis Default Reports
Incremental synthesis: None

Implementation

Status: Not s
Messages: No e
Part: xc7z(
Strategy: Vivac
Report Strategy: Vivac
Incremental implementation: None

h. The following prompt appears. Select “Add or Create design sources”. Click “Next”.



Add Sources

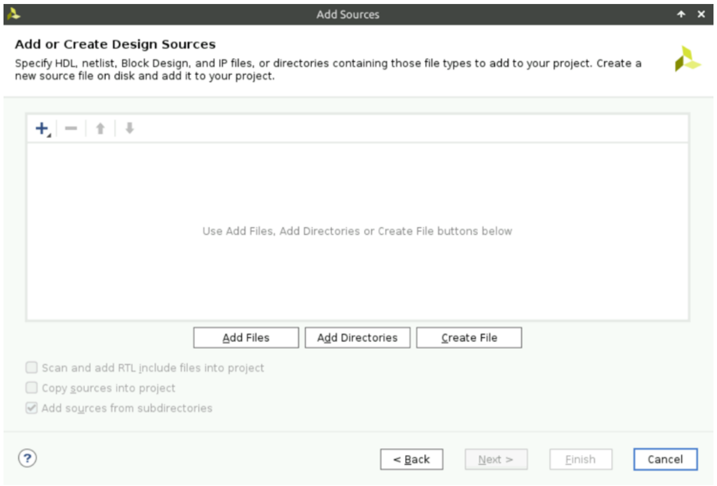
This guides you through the process of adding and creating sources for your project

○ Add or create constraints

● Add or create design sources

○ Add or create simulation sources

i. Click on “Create File”.



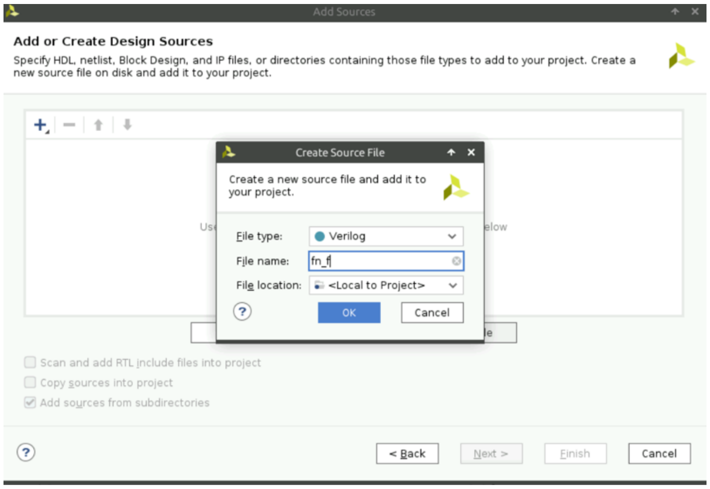
Add or Create Design Sources

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

☐ Scan and add RTL include files into project
☐ Copy sources into project
☑ Add sources from subdirectories

j. Name your file “fn_f” and choose Verilog as file type. Click OK.



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 a new source file and add it to your project.

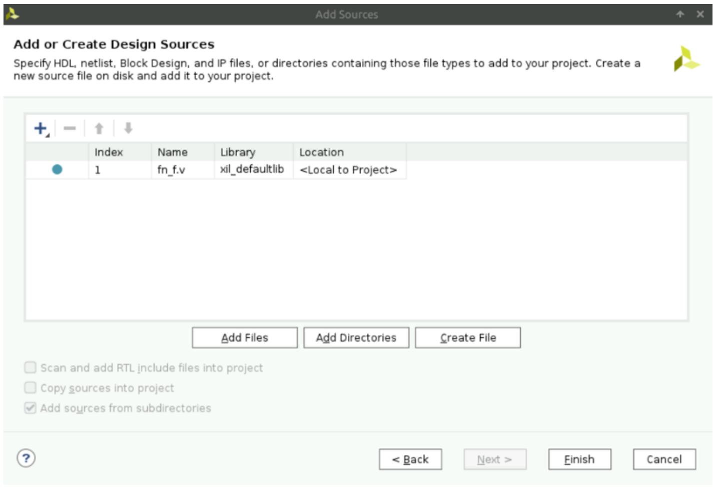
File type: ● Verilog

File name: fn_f

File location: <Local to Project>

☐ Scan and add RTL include files into project
☐ Copy sources into project
☑ Add sources from subdirectories

k. Click Finish to add the source to your project.



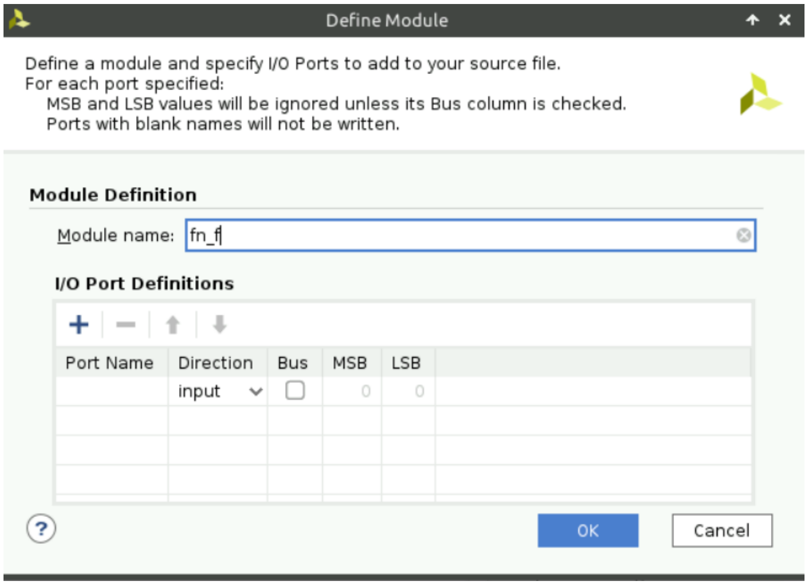
Add or Create Design Sources

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	fn_f.v	xil_defaultlib	<Local to Project>

☐ Scan and add RTL include files into project
☐ Copy sources into project
☑ Add sources from subdirectories

l. Click ok to define the module.



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.

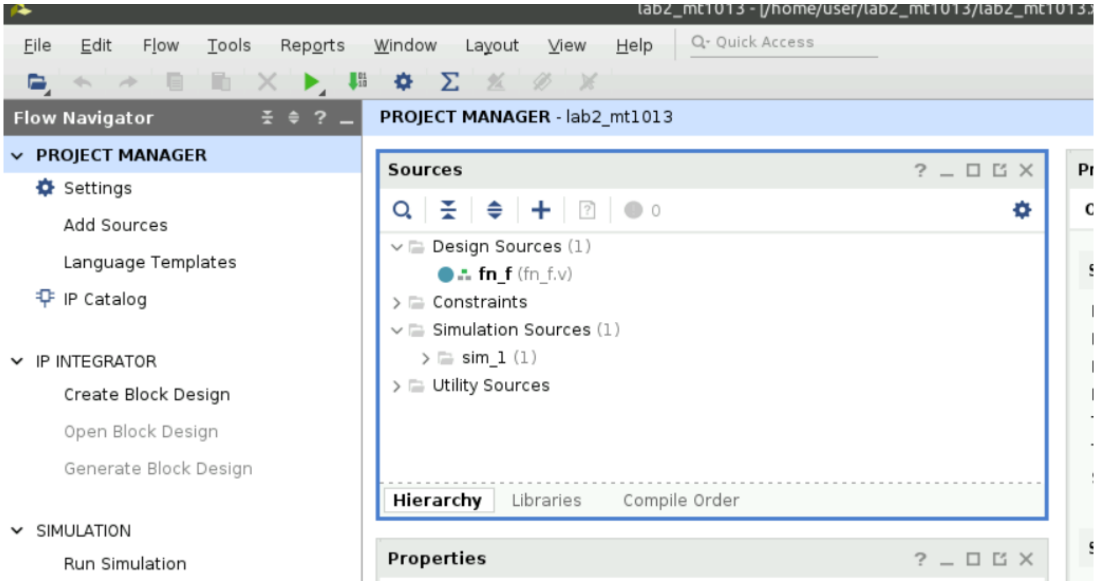
Module Definition

Module name: fn_f

I/O Port Definitions

Port Name	Direction	Bus	MSB	LSB
	input	☐	0	0

m. Locate the source "fn_f" in the project. Open "fn_f" by double clicking on the module.



lab2_mt1013 - [/home/user/lab2_mt1013/lab2_mt1013.

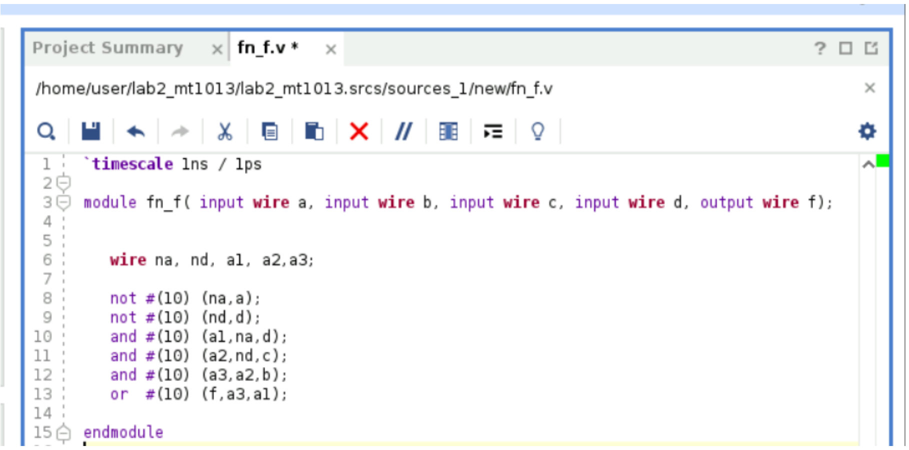
PROJECT MANAGER - lab2_mt1013

Sources

- Design Sources (1)
  - fn_f (fn_f.v)
- Constraints
- Simulation Sources (1)
  - sim_1 (1)
- Utility Sources

3. Copy and paste the following code in “fn_f” source.

```
`timescale 1ns / 1ps

module fn_f( input wire a, input wire b, input wire c, input wire d, output wire f);

  wire na, nd, a1, a2,a3;

  not #(10) (na,a);
  not #(10) (nd,d);
  and #(10) (a1,na,d);
  and #(10) (a2,nd,c);
  and #(10) (a3,a2,b);
  or  #(10) (f,a3,a1);

endmodule
```

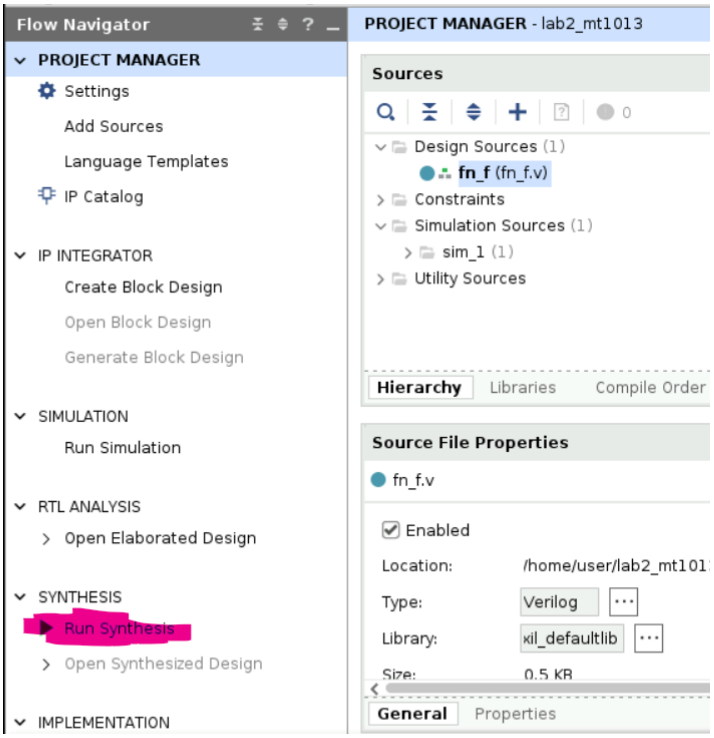


/home/user/lab2_mt1013/lab2_mt1013.srcs/sources_1/new/fn_f.v

```
`timescale 1ns / 1ps

module fn_f( input wire a, input wire b, input wire c, input wire d, output wire f);


    wire na, nd, a1, a2,a3;

    not #(10) (na,a);
    not #(10) (nd,d);
    and #(10) (a1,na,d);
    and #(10) (a2,nd,c);
    and #(10) (a3,a2,b);
    or  #(10) (f,a3,a1);

endmodule
```

4. Run Synthesis from the flow navigator.

s



☑ Enabled

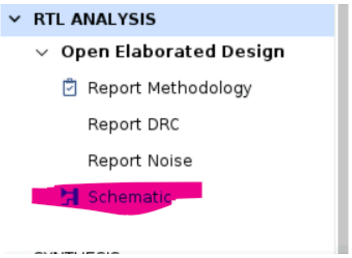
Location: /home/user/lab2_mt101

Type: Verilog

Library: xil_defaultlib

Size: 0.5 KB

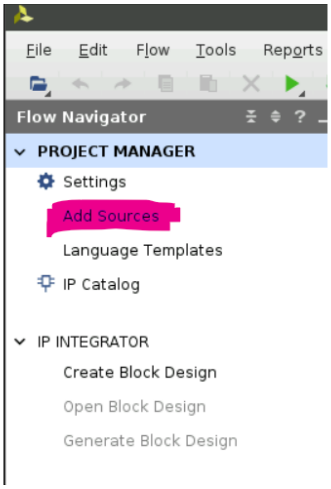
5. Open the elaborated design under the RTL analysis tab. Submit the Elaborated design schematic diagram with your prelab report. This diagram will contain the schematic for the function $F(A, B, C, D)$ in the form of inverters, AND and OR gates. Include this diagram in your prelab report. How many gates are used for the implementation of this function?



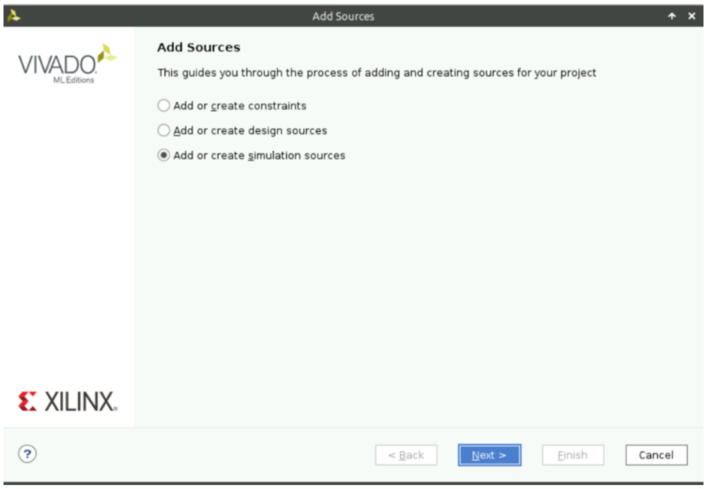
6. Create a simulation file in Vivado and use the following code for running the simulation for each possible combinations of the input. The detailed instructions can be found below.

```
`timescale 1ns / 1ps

module f_sim(
    );

    reg a;
    reg b;
    reg c;
    reg d;

    wire f;

    fn_f myfunc(a,b,c,d,f);

    initial
    begin

    { a,b,c,d } = 4'd0;
    # 100 { a,b,c,d } = 4'd1;
    # 100 { a,b,c,d } = 4'd2;
    # 100 { a,b,c,d } = 4'd3;
    # 100 { a,b,c,d } = 4'd4;
    # 100 { a,b,c,d } = 4'd5;
    # 100 { a,b,c,d } = 4'd6;
    # 100 { a,b,c,d } = 4'd7;
    # 100 { a,b,c,d } = 4'd8;
    # 100 { a,b,c,d } = 4'd9;
    # 100 { a,b,c,d } = 4'd10;
    # 100 { a,b,c,d } = 4'd11;
    # 100 { a,b,c,d } = 4'd12;
    # 100 { a,b,c,d } = 4'd13;
    # 100 { a,b,c,d } = 4'd14;
    # 100 { a,b,c,d } = 4'd15;

    end
endmodule
```

a. Click on “Add Sources” from the Flow Navigator.



b. Choose “Add or create simulation sources”. Click “Next”.



Add Sources

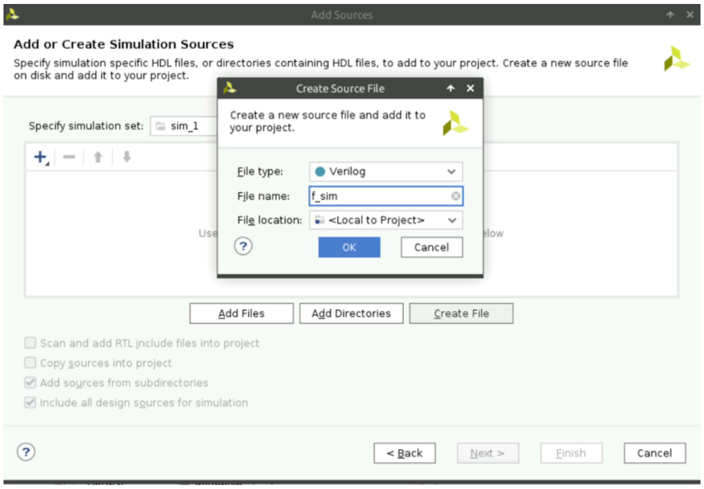
This guides you through the process of adding and creating sources for your project

○ Add or create constraints

○ Add or create design sources

● Add or create simulation sources

c. Name the file “f_sim” and Choose Verilog as the file type.



Add or Create Simulation Sources

Specify simulation specific HDL files, or directories containing HDL files, to add to your project. Create a new source file on disk and add it to your project.

Specify simulation set: sim_1

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

File type: Verilog

File name: f_sim

File location: <Local to Project>

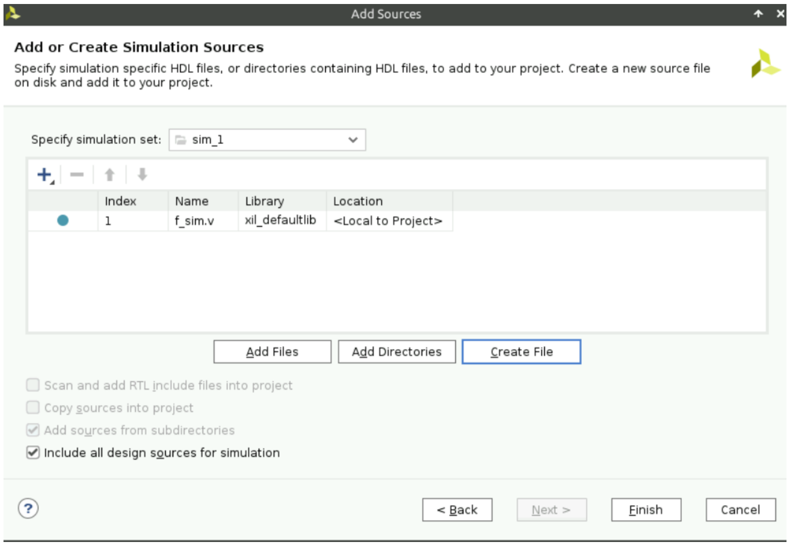
□ Scan and add RTL include files into project

□ Copy sources into project

☑ Add sources from subdirectories

☑ Include all design sources for simulation

d. Click “Finish” to add the simulation file to the project.



Add Sources

Add or Create Simulation Sources

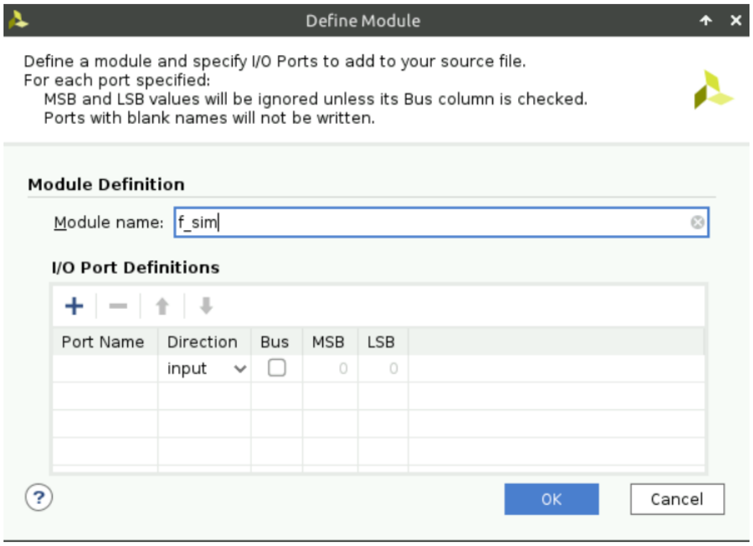
Specify simulation specific HDL files, or directories containing HDL files, to add to your project. Create a new source file on disk and add it to your project.

Specify simulation set: sim_1

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

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

e. Click OK to define the module.



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.

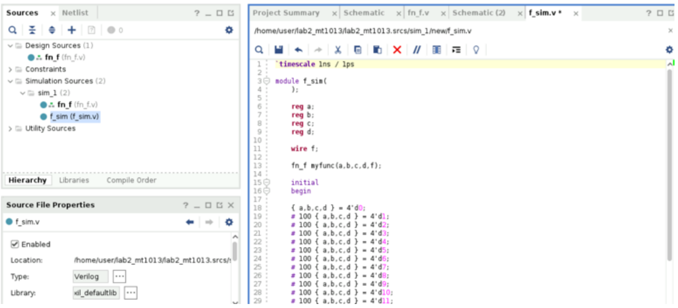
Module Definition

Module name: f_sim

I/O Port Definitions

Port Name	Direction	Bus	MSB	LSB
	input	☐	0	0

f. Locate the source “f_sim” and paste the provided code.



☑ Enabled

Location: /home/user/lab2_mt1013/lab2_mt1013.srcs/

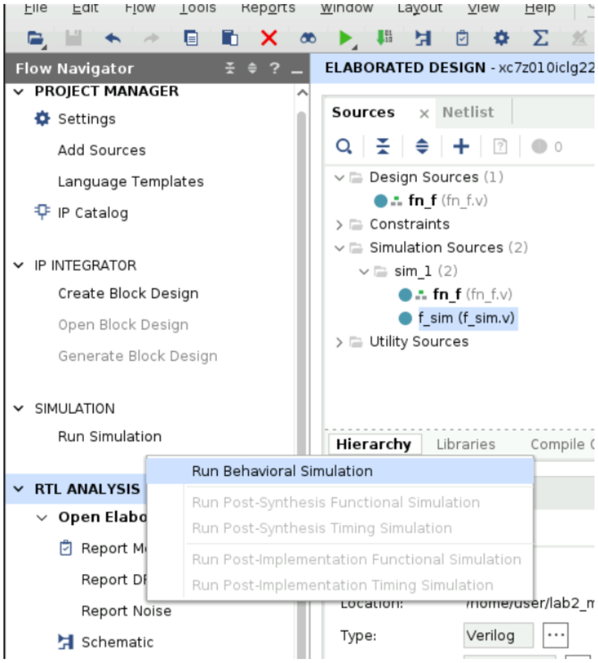
Type: Verilog

Library: xil_defaultlib

/home/user/lab2_mt1013/lab2_mt1013.srcs/sim_1/new/f_sim.v

```
`timescale 1ns / 1ps

module f_sim(
    );

    reg a;
    reg b;
    reg c;
    reg d;

    wire f;

    fn_f myfunc(a,b,c,d,f);

    initial
    begin

    { a,b,c,d } = 4'd0;
    # 100 { a,b,c,d } = 4'd1;
    # 100 { a,b,c,d } = 4'd2;
    # 100 { a,b,c,d } = 4'd3;
    # 100 { a,b,c,d } = 4'd4;
    # 100 { a,b,c,d } = 4'd5;
    # 100 { a,b,c,d } = 4'd6;
    # 100 { a,b,c,d } = 4'd7;
    # 100 { a,b,c,d } = 4'd8;
    # 100 { a,b,c,d } = 4'd9;
    # 100 { a,b,c,d } = 4'd10;
    # 100 { a,b,c,d } = 4'd11;
```

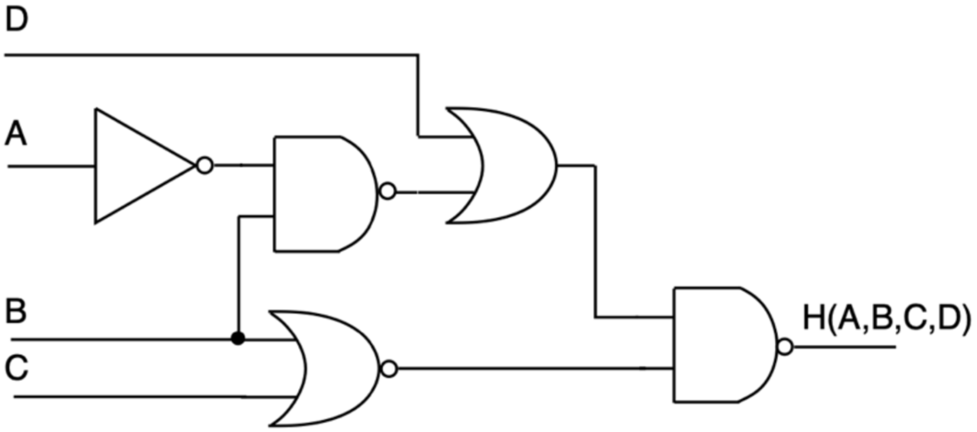
7. Simulate the testbench and obtain the timing diagram which demonstrates each of the possible input combinations along with the corresponding output. Click on "Run Simulation" from the flow navigator and choose "Run Behavioral Simulation". Run the simulation for at least 2 µs. Include the waveform in your prelab report.



8. There is a timing hazard in the circuit. Identify when it occurs. Discuss in your prelab report.

1.2 Analysis of A Simple Combinational Circuit

Consider the circuit shown below, that generates the Boolean function $H(A, B, C, D)$. Find the Boolean expression for this function, and complete the truth table. Include these in your prelab report.



A	B	C	D	$\mathrm{H(A, B, C, D)}$
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

2.Experiments

2.1 Synthesizing A Hazard-free Circuit

1. Build the ICOD that you used in Lab1. Check that the ICOD on your breadboard is working properly, by toggling each of the input pins from logic '0' (GND) to logic '1' (5V), and from logic '1' (5V) to logic '0' (GND).
2. Use the Karnaugh map to find a version of $\mathrm{F(A, B, C, D)}$ that does not have timing hazard. Let's name this function $\mathrm{G(A, B, C, D)}$.
3. Draw the NAND-NAND implementation of $\mathrm{G(A, B, C, D)}$.
4. Implement the timing hazard free circuit $\mathrm{G(A, B, C, D)}$.
5. Connect the A, B,C, and D inputs to four switches, and the output $\mathrm{G(A, B, C, D)}$ to a display unit (LED). Generate all possible inputs listed in the truth table, and verify that the output follows the logic function listed in the truth table. If it doesn't, practice to find the issue and debug the circuit.
6. Show your TA the circuit and verify some truth table values in his/her presence.

A	B	C	D	$G(A,B,C,D)$
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

2.2 Analyzing A Combinational Circuit

1. Implement the circuit $H(A,B,C,D)$ using the ICs available in the lab.
2. Connect the A, B,C, and D inputs to four switches, and the output $H(A,B,C,D)$ to a display unit (LED). Set the inputs B and C to logic "0" and complete the table below. Show your circuit and the completed table to your TA.

A	D	$H(A,0,0,D)$
0	0	
0	1	
1	0	
1	1	

3. Lab 2 Report

Your report for Lab 2 should include the following:

- your prelab report with all the materials requested.

- Karnaugh maps and logic diagram for function $G(A, B, C, D)$. Discussion on how you made the circuit free of timing hazards.

- Images of the circuits that you implemented to realize functions $G(A, B, C, D)$ and $H(A, B, C, D)$.

Additional Resources for Verliog and Vivado

The Following resources have been posted on Canvas :

i. Vivado Tutorial.pdf
ii. Vivado-How to Run.pdf
iii. FPGA Compiler.pdf
iv. Intro to Verilog.pdf
v. Verilog_Reference_Guide.pdf