# ELD Lab 1
# Design of Full Adder

# Objective

- Design and implement a 4-bit adder for unsigned inputs using Full Adder
- Write the suitable testbench and verify the functionality of the full adder.
- **Lab Homework:** Extend the design to 4-bit adder/subtractor circuit for signed numbers

# Theory

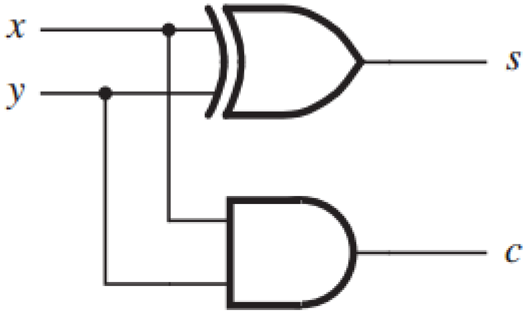
# HA (single bit) and FA (multi-bit)

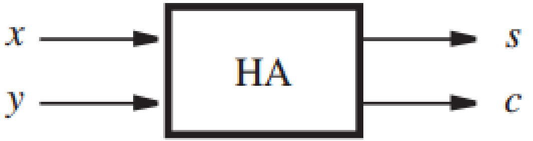
| | | Carry | Sum |
|---|---|---|---|
| $x$ | $y$ | $c$ | $s$ |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

$$c_{i+1} = \bar{c}_i \cdot x_i \cdot y_i + c_i \cdot \bar{x}_i \cdot y_i + c_i \cdot x_i \cdot \bar{y}_i + c_i \cdot x_i \cdot y_i$$

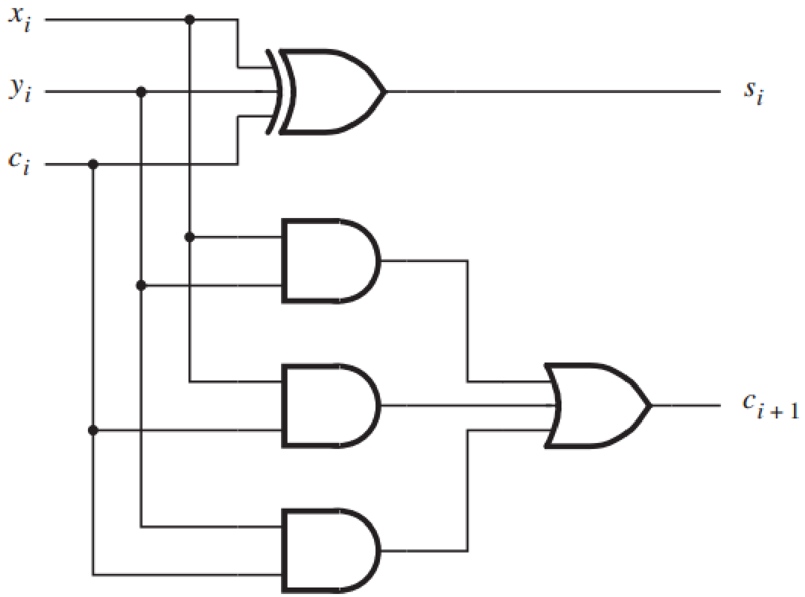
$$c_{i+1} = x_i \cdot y_i + c_i \cdot (\bar{x}_i \cdot y_i + x_i \cdot \bar{y}_i)$$

$$c_{i+1} = x_i \cdot y_i + c_i \cdot (x_i \oplus y_i)$$





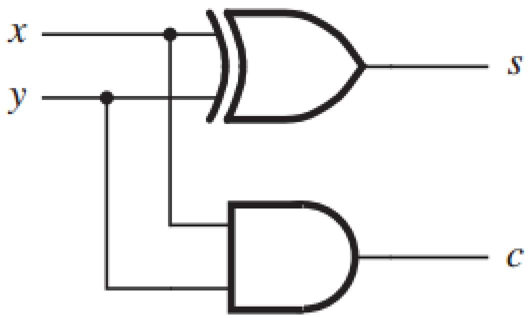
| $c_i$ | $x_i$ | $y_i$ | $c_{i+1}$ | $s_i$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

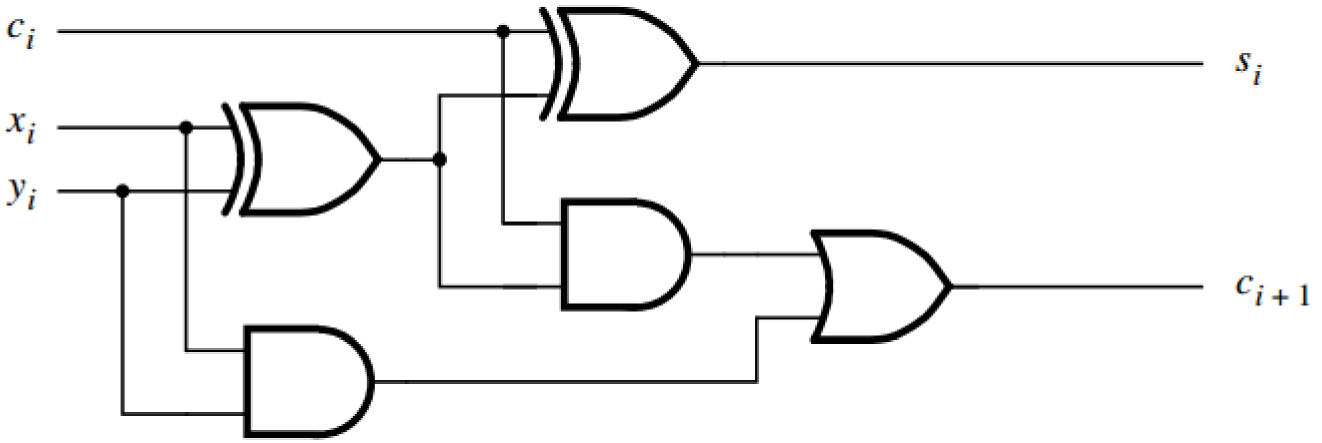


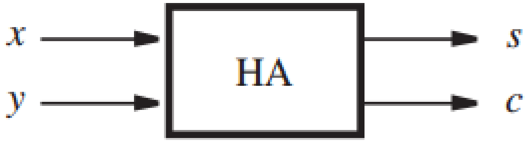
# HA (single bit) and FA (multi-bit)

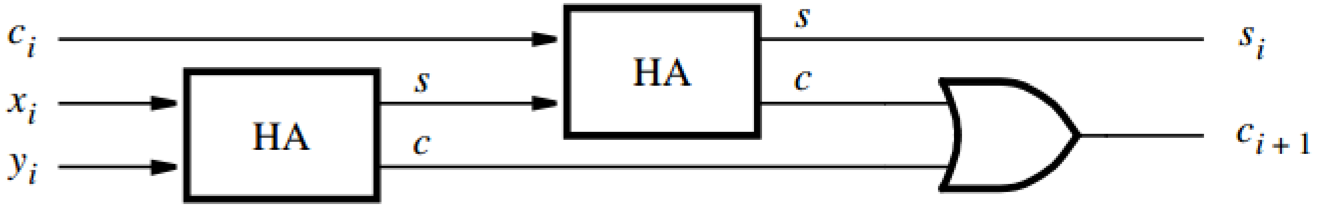
$$s_i = (x_i \oplus y_i \oplus c_i)$$

$$c_{i+1} = x_i \cdot y_i + c_i \cdot (x_i \oplus y_i)$$



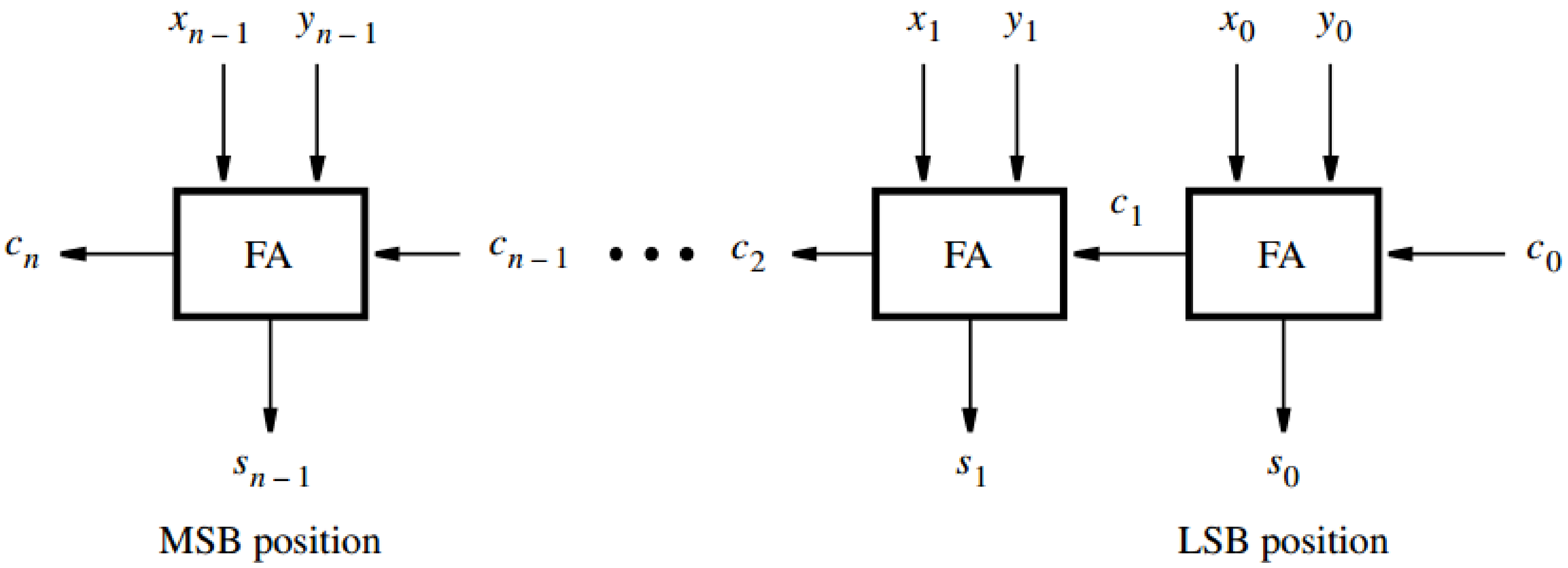




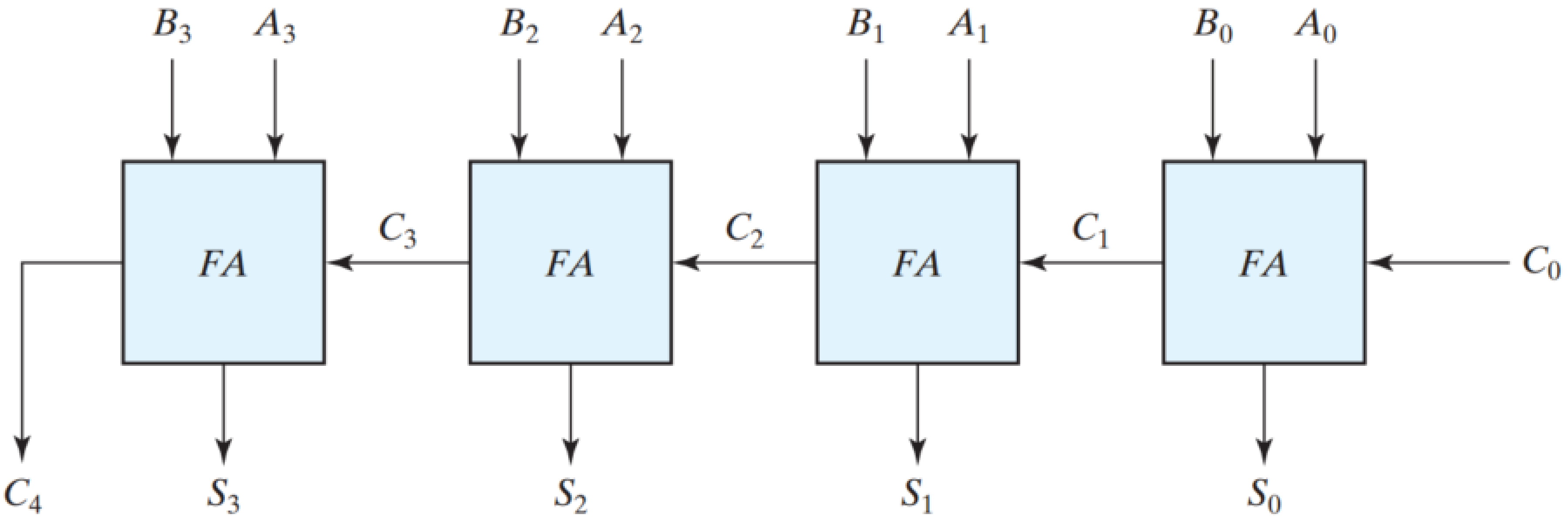


**This approach minimizes the number of ICs needed to implement the circuit, and it reduces the wiring complexity substantially.**

# 4-bit FA Block diagram

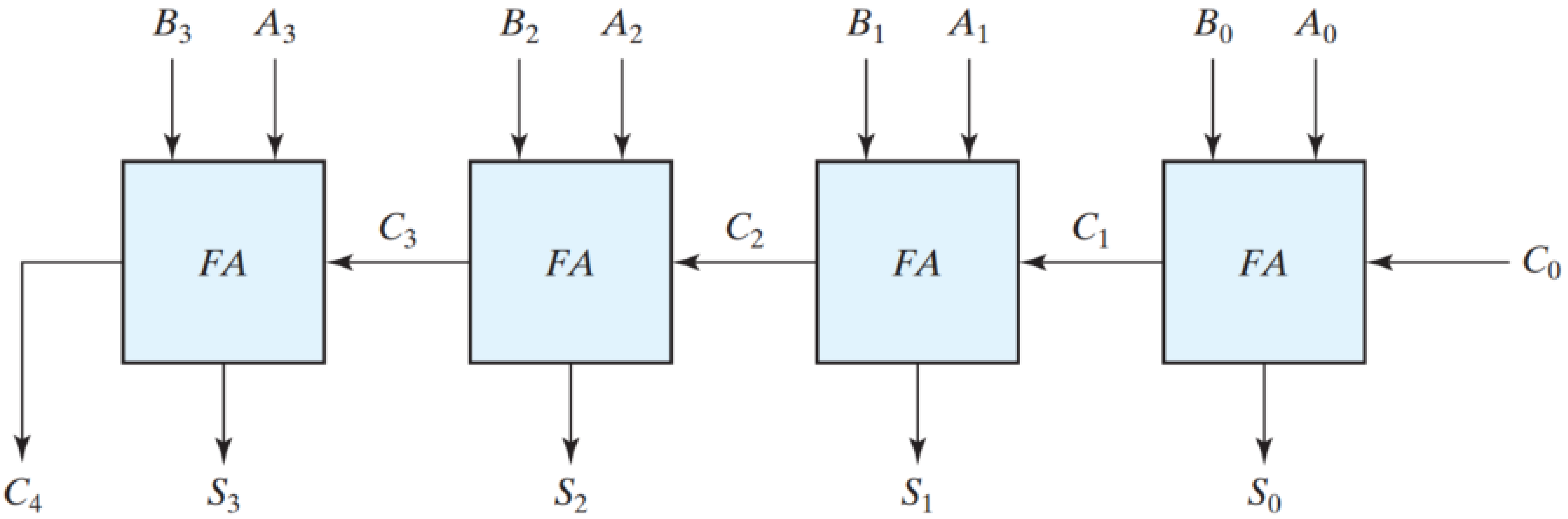


# Adders



- For adder with unsigned number inputs, add output flag which goes high when overflow occurs
- When two numbers with *n* digits each are added and the sum is a number occupying *n* + 1 digits, we say that an overflow occurred.
- Overflow is a problem in digital computers because the number of bits that hold the number is finite and a result that contains *n* + 1 bits cannot be accommodated by an *n* -bit word.

# Adders

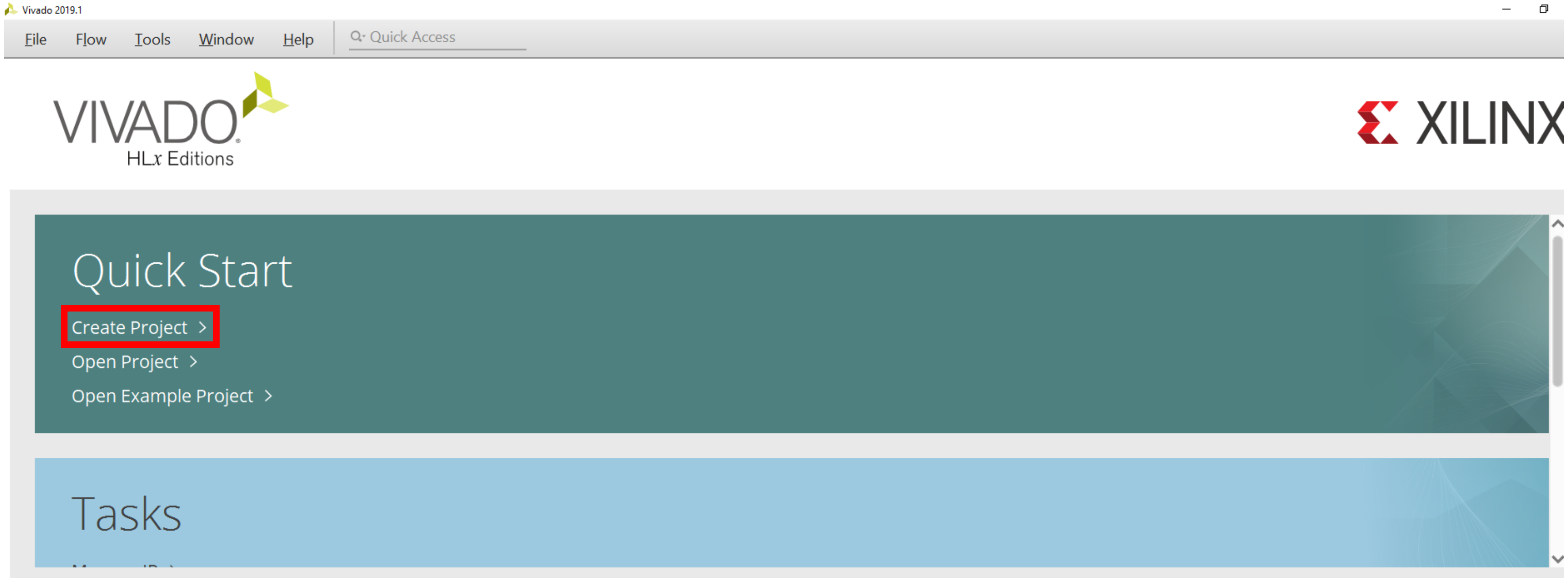


- For adder with unsigned number inputs, add output flag which goes high when overflow occurs
- When two unsigned numbers are added, an overflow is detected from the end carry out of the most significant position.

# Lab

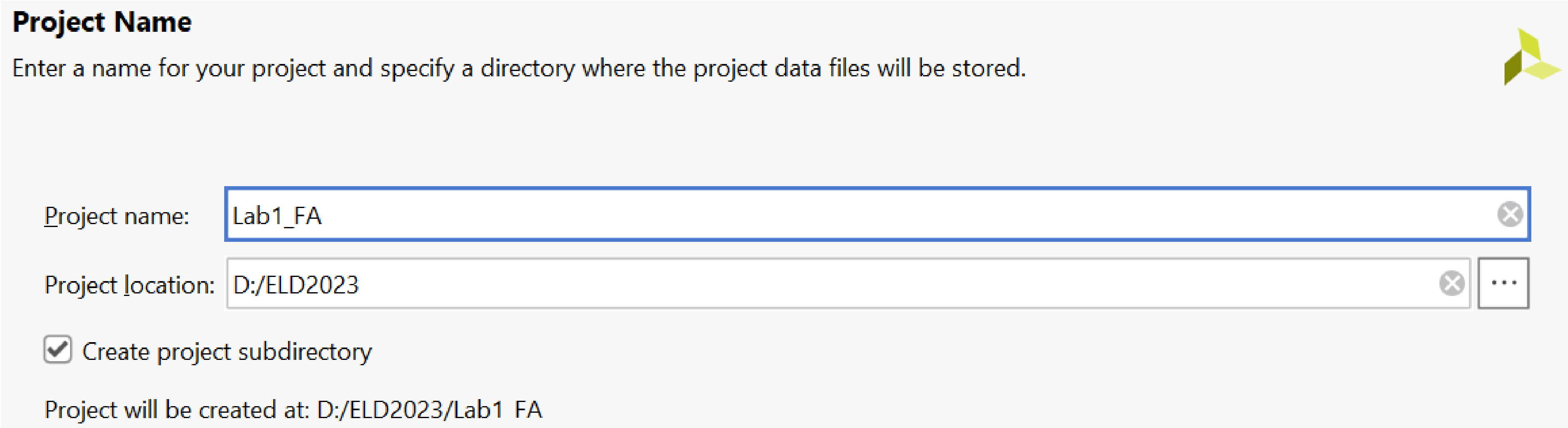
# Open the Vivado

- Select Create Project and click on Next



# Open the Vivado

- Select appropriate project folder.
- Avoid windows folder and space in address



**Project Name**

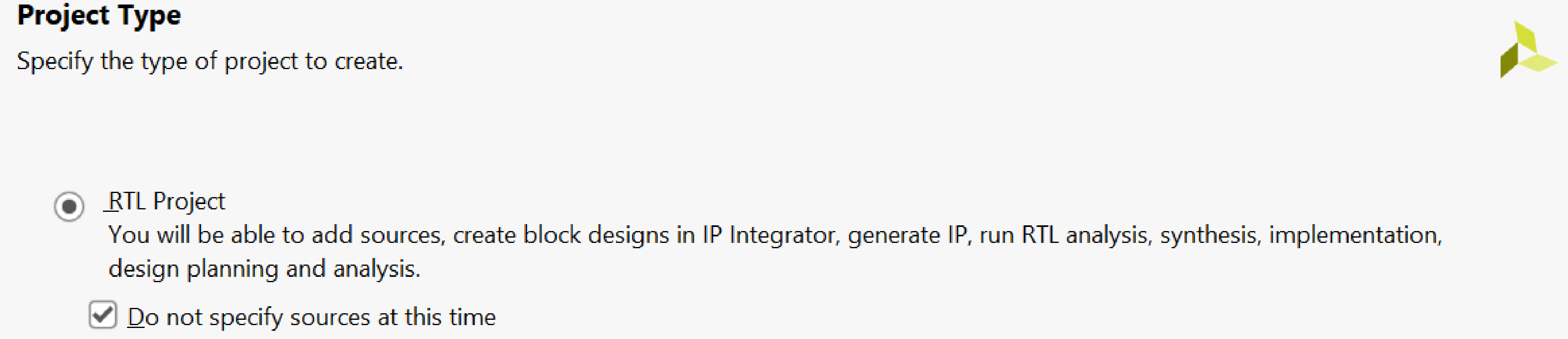
Enter a name for your project and specify a directory where the project data files will be stored.

Project name: Lab1_FA

Project location: D:/ELD2023

☑ Create project subdirectory

Project will be created at: D:/ELD2023/Lab1 FA



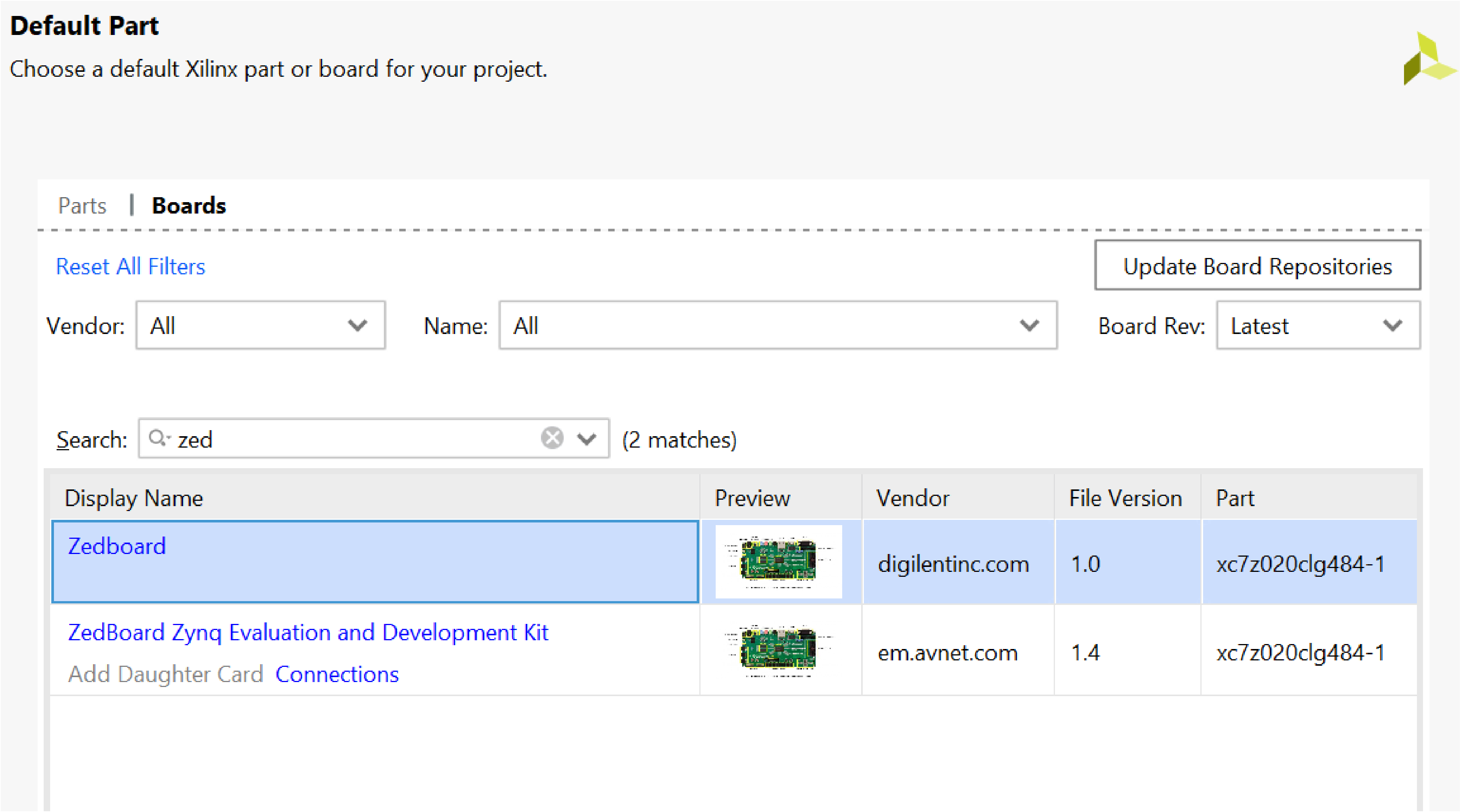
**Project Type**

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

# Select Zedboard



**Default Part**

Choose a default Xilinx part or board for your project.

Parts | **Boards**

Reset All Filters

Vendor: All

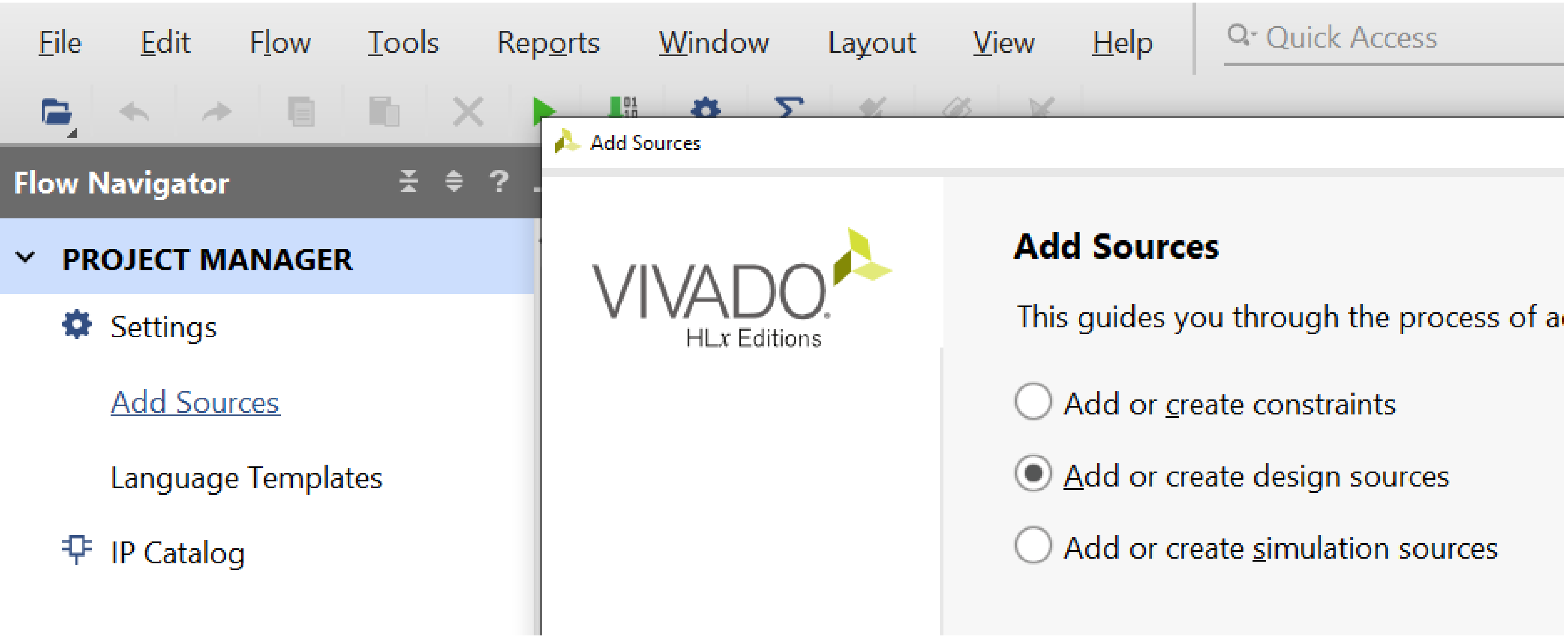
Name: All

Board Rev: Latest

Search: zed (2 matches)

| Display Name | Preview | Vendor | File Version | Part |
|---|---|---|---|---|
| Zedboard | | digilentinc.com | 1.0 | xc7z020clg484-1 |
| ZedBoard Zynq Evaluation and Development Kit<br>Add Daughter Card Connections | | em.avnet.com | 1.4 | xc7z020clg484-1 |

# Vivado: Project Manager



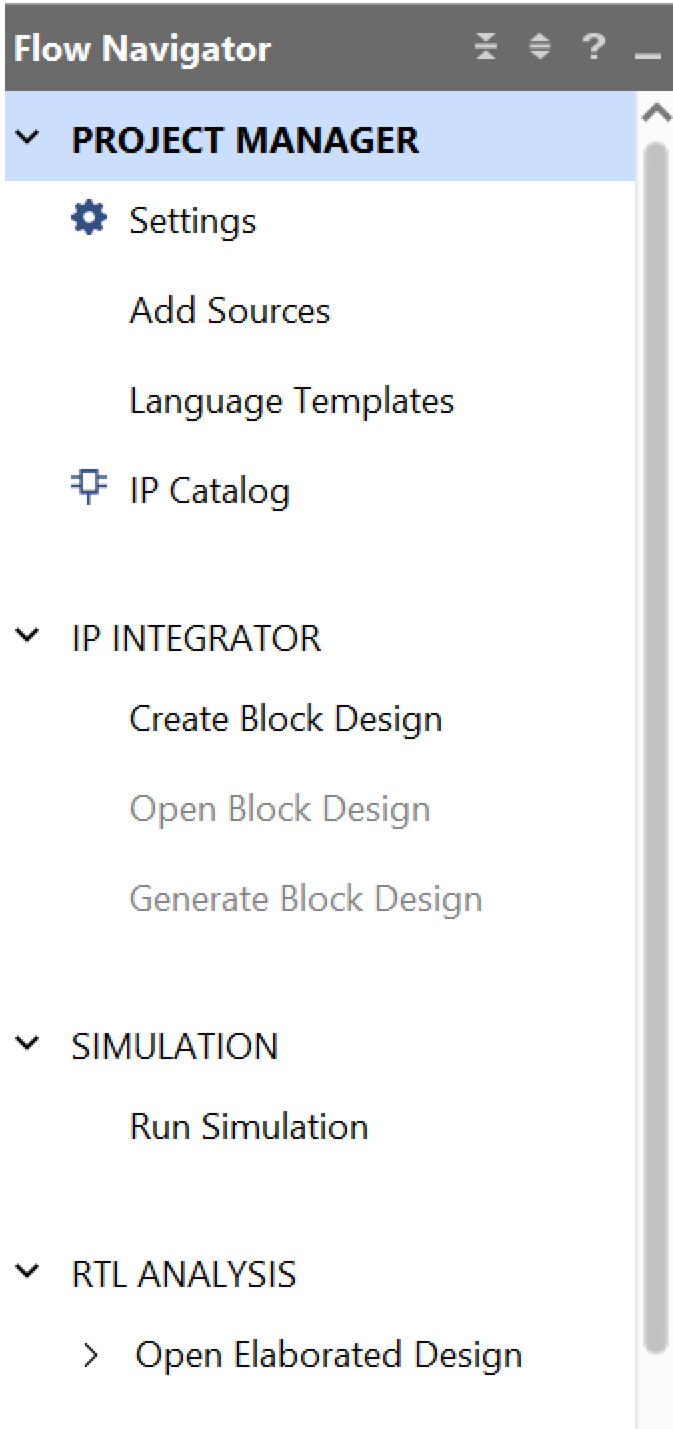
**Add Sources**

This guides you through the process of a

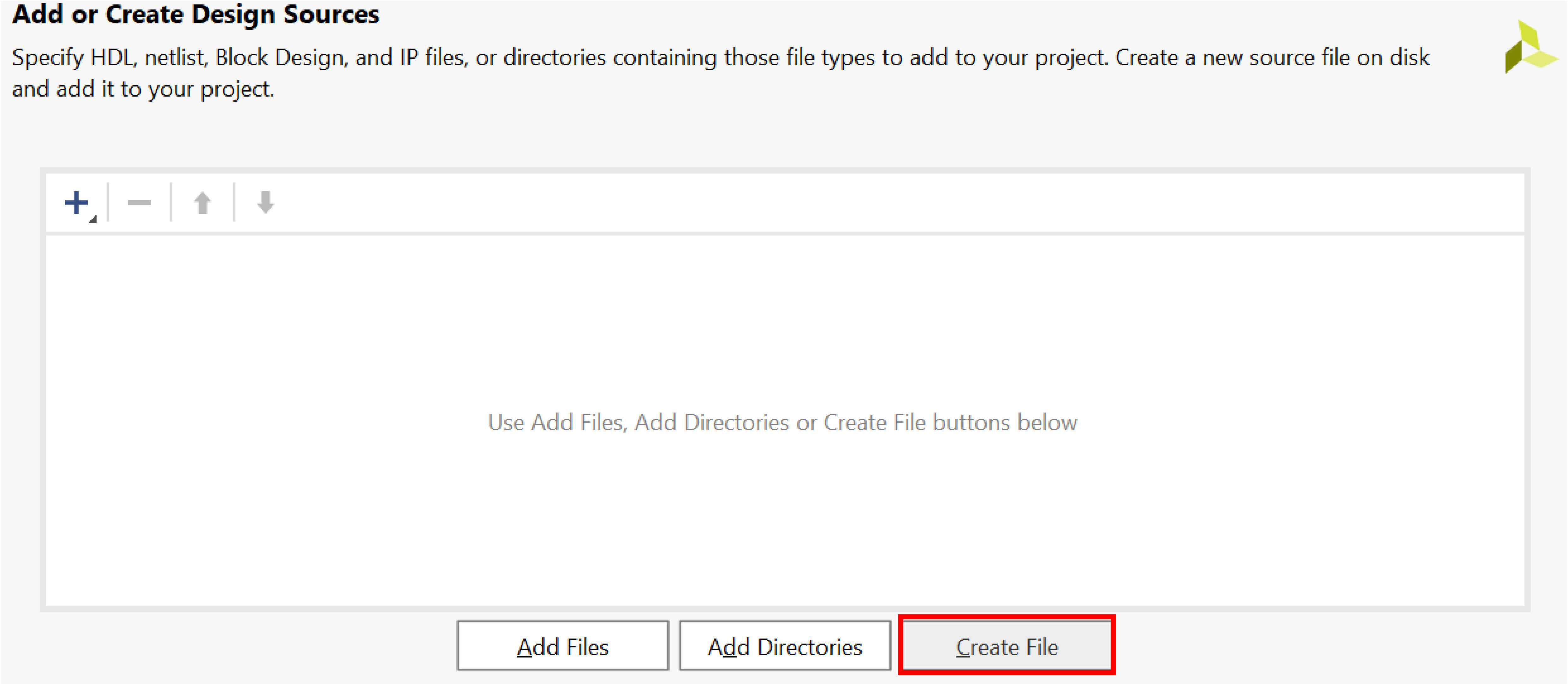
○ Add or create constraints

● Add or create design sources

○ Add or create simulation sources



# Design Sources

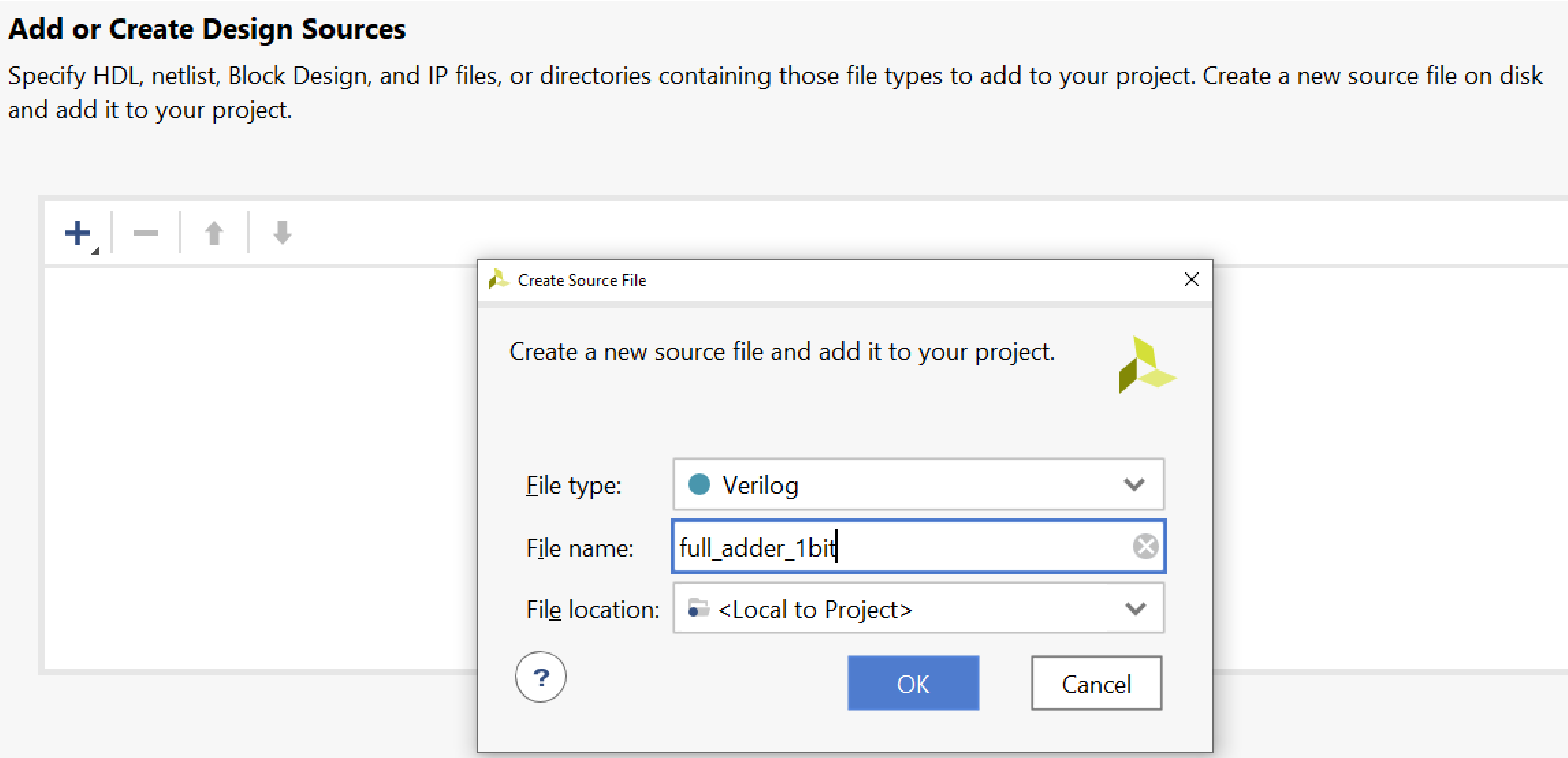


**Add or Create Design Sources**

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

# Design 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.

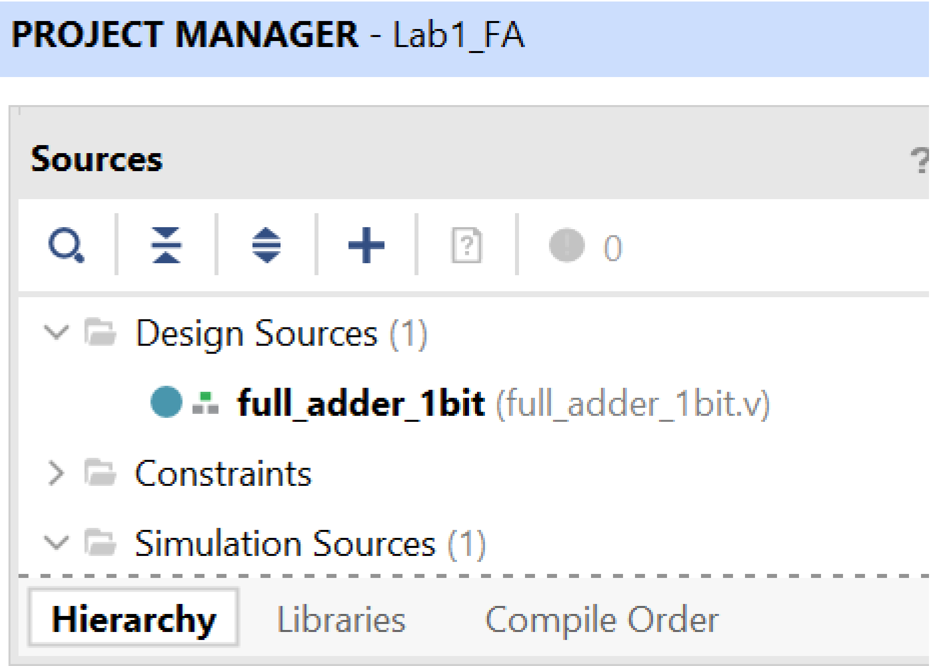
Create a new source file and add it to your project.

File type: ● Verilog

File name: full_adder_1bit

File location: <Local to Project>

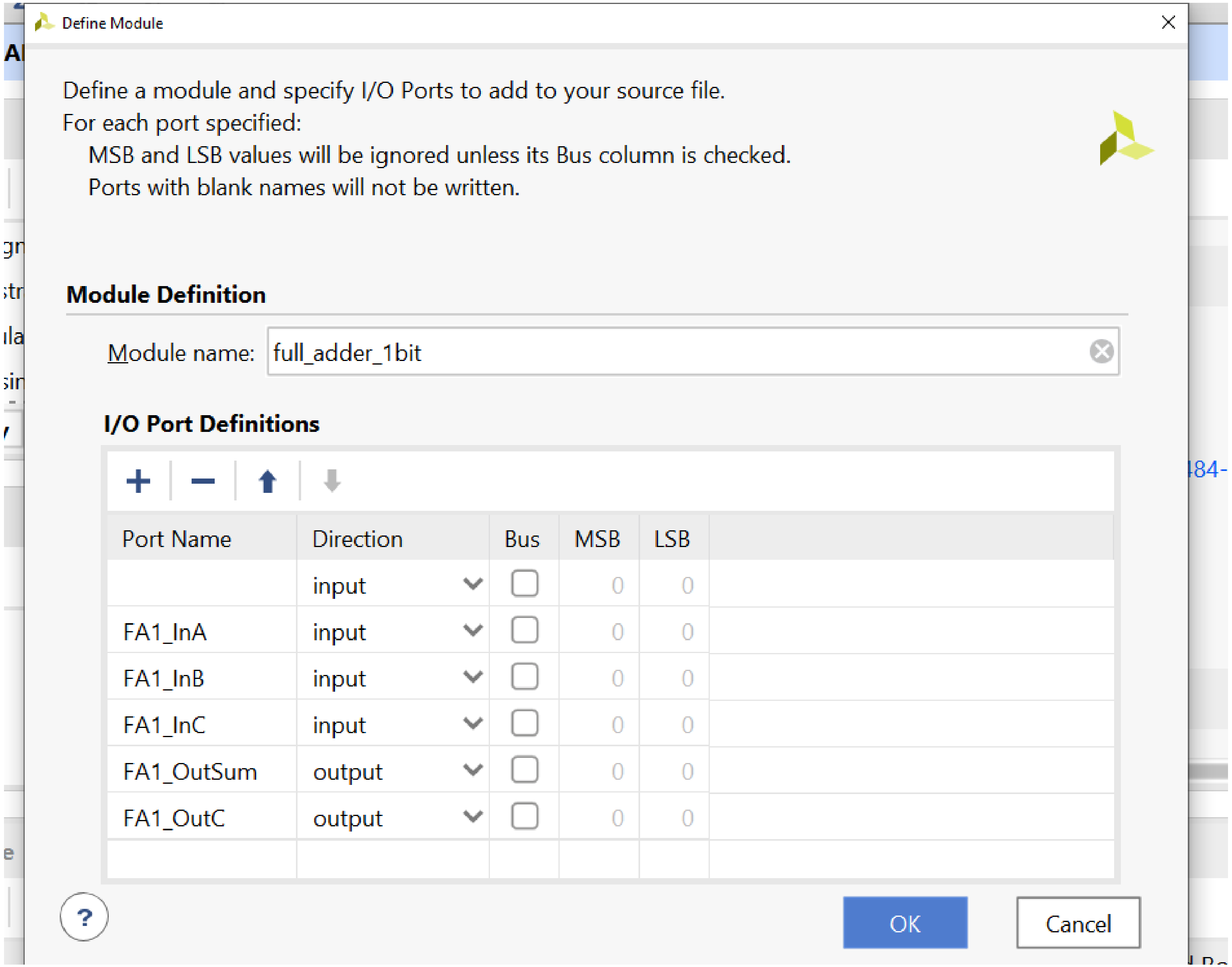
# Design Sources



**PROJECT MANAGER** - Lab1_FA

**Sources**

- Design Sources (1)
  - full_adder_1bit (full_adder_1bit.v)
- Constraints
- Simulation Sources (1)



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.

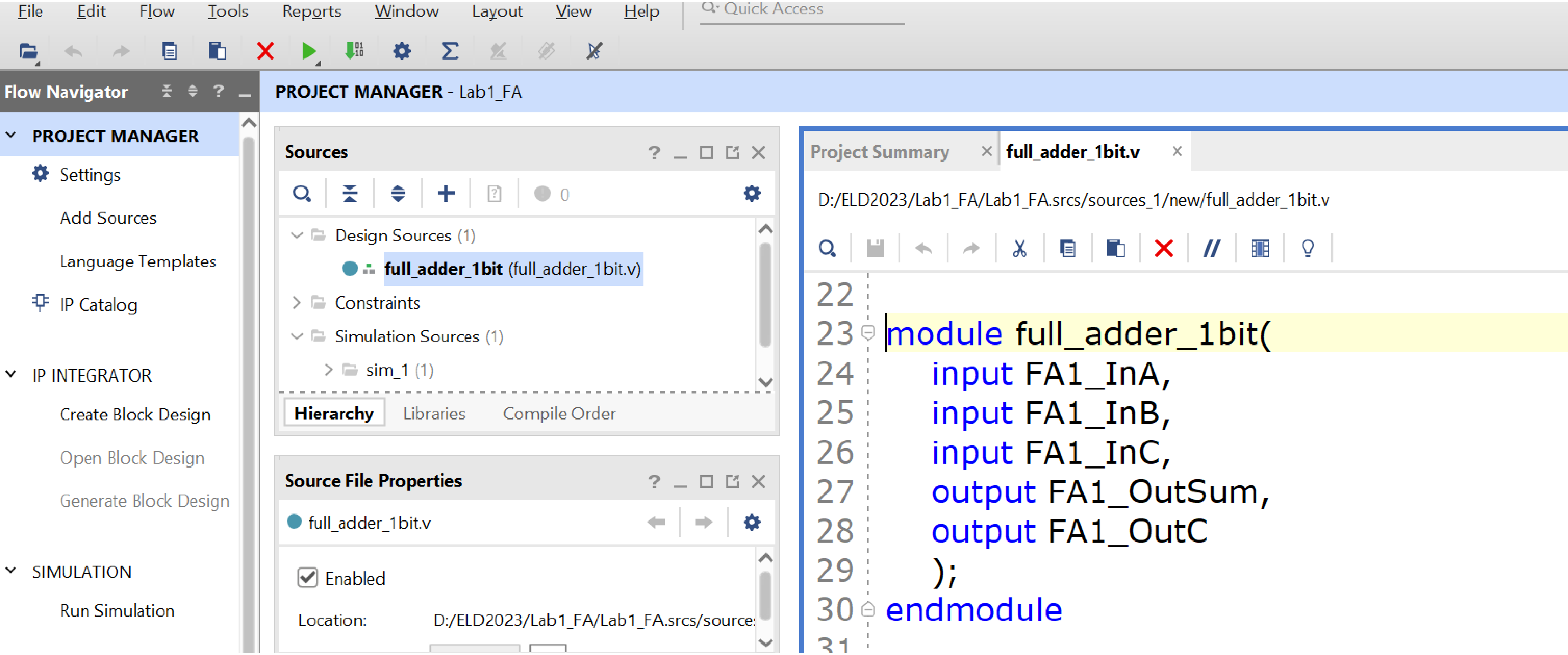
**Module Definition**

Module name: full_adder_1bit

**I/O Port Definitions**

| Port Name | Direction | Bus | MSB | LSB |
|---|---|---|---|---|
|  | input | □ | 0 | 0 |
| FA1_InA | input | □ | 0 | 0 |
| FA1_InB | input | □ | 0 | 0 |
| FA1_InC | input | □ | 0 | 0 |
| FA1_OutSum | output | □ | 0 | 0 |
| FA1_OutC | output | □ | 0 | 0 |

# Design Sources



Design Sources (1)
- full_adder_1bit (full_adder_1bit.v)

Constraints

Simulation Sources (1)
- sim_1 (1)

full_adder_1bit.v

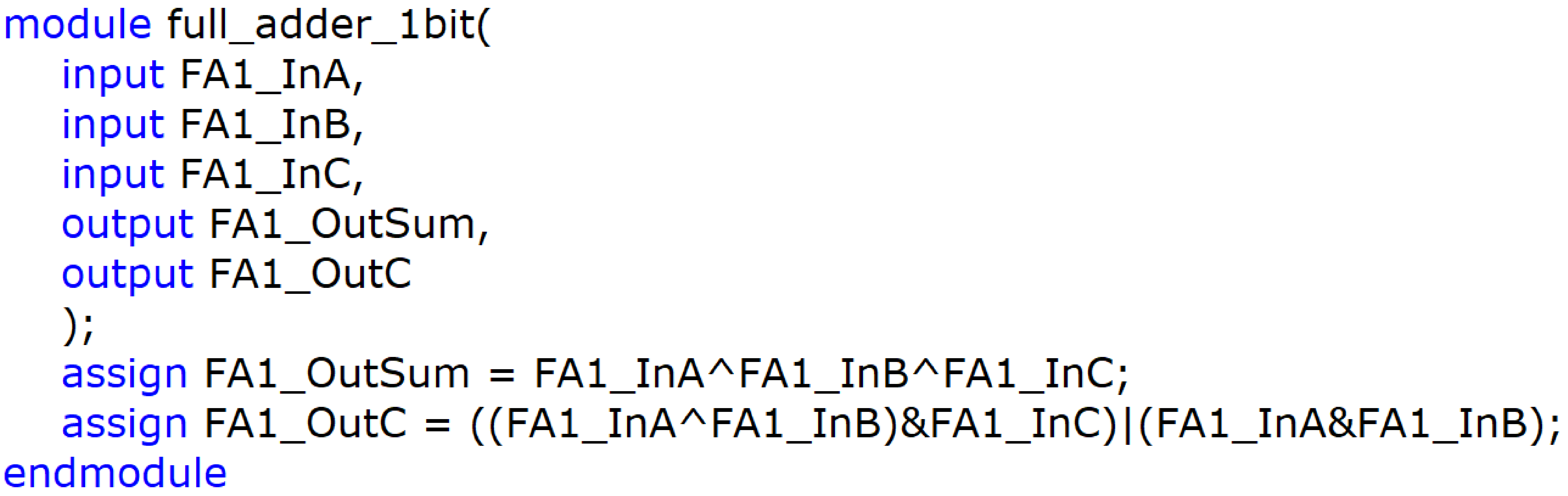
☑ Enabled

Location: D:/ELD2023/Lab1_FA/Lab1_FA.srcs/source

D:/ELD2023/Lab1_FA/Lab1_FA.srcs/sources_1/new/full_adder_1bit.v

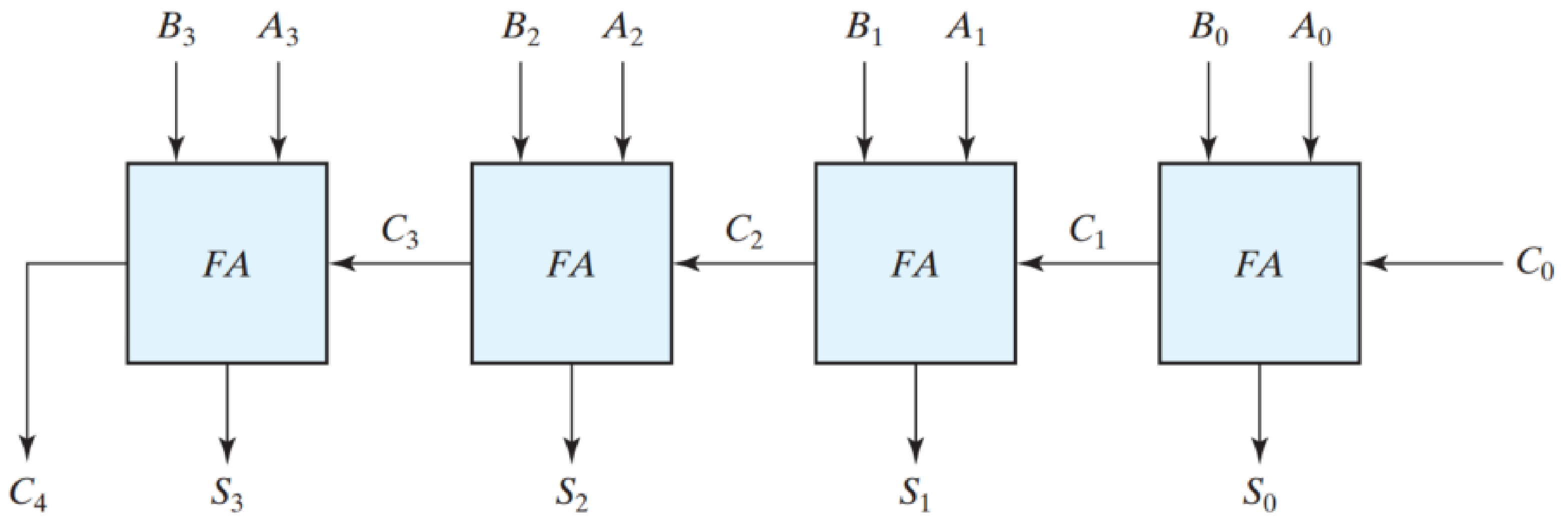
```
module full_adder_1bit(
    input FA1_InA,
    input FA1_InB,
    input FA1_InC,
    output FA1_OutSum,
    output FA1_OutC
    );
endmodule
```

# Design Sources



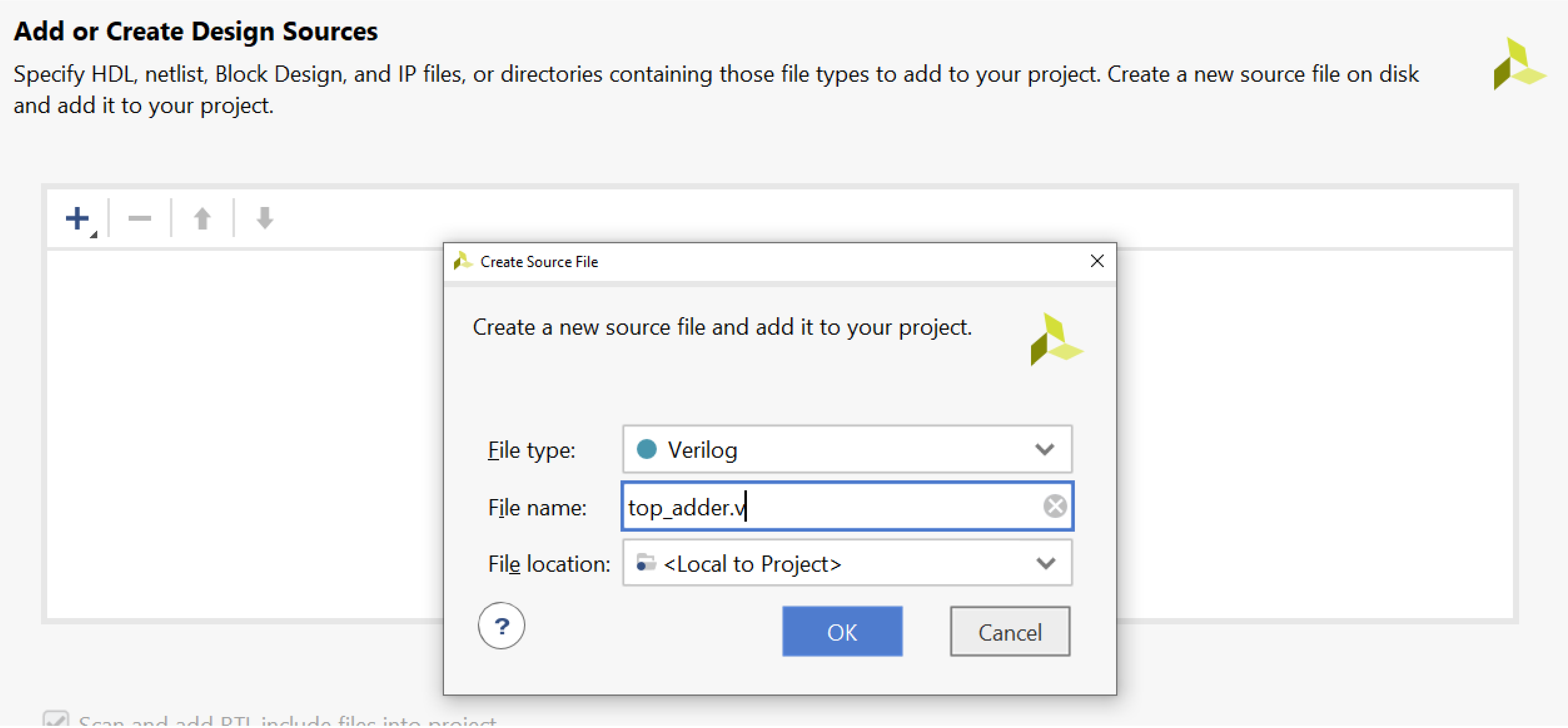
```
module full_adder_1bit(
    input FA1_InA,
    input FA1_InB,
    input FA1_InC,
    output FA1_OutSum,
    output FA1_OutC
    );
    assign FA1_OutSum = FA1_InA^FA1_InB^FA1_InC;
    assign FA1_OutC = ((FA1_InA^FA1_InB)&FA1_InC)|(FA1_InA&FA1_InB);
endmodule
```

# 4-bit FA



- Add new source file top_adder.v

# 4-bit FA



**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.

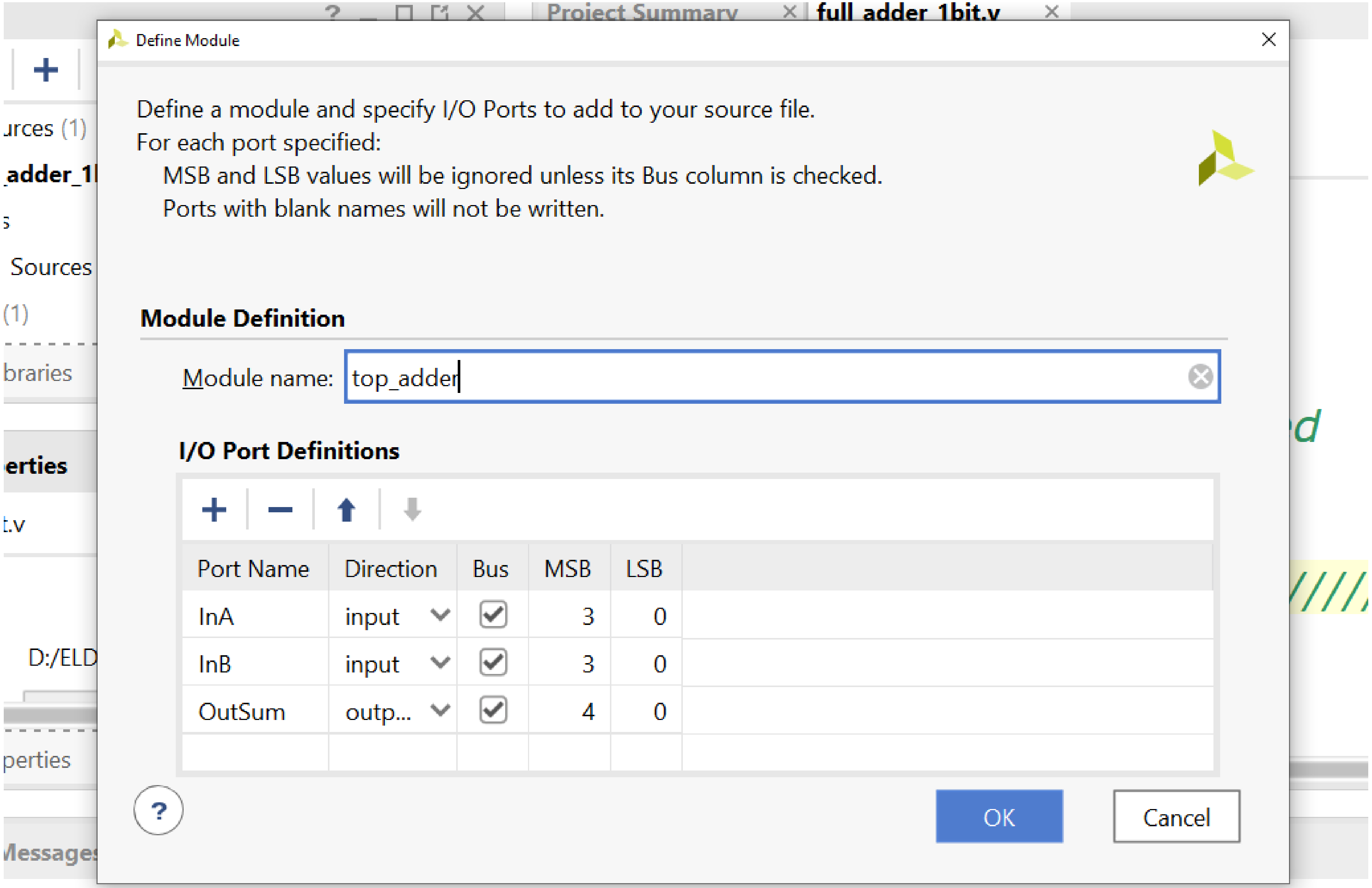
File type: ● Verilog

File name: top_adder.v

File location: <Local to Project>

☑ Scan and add RTL include files into project

# 4-bit FA



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.

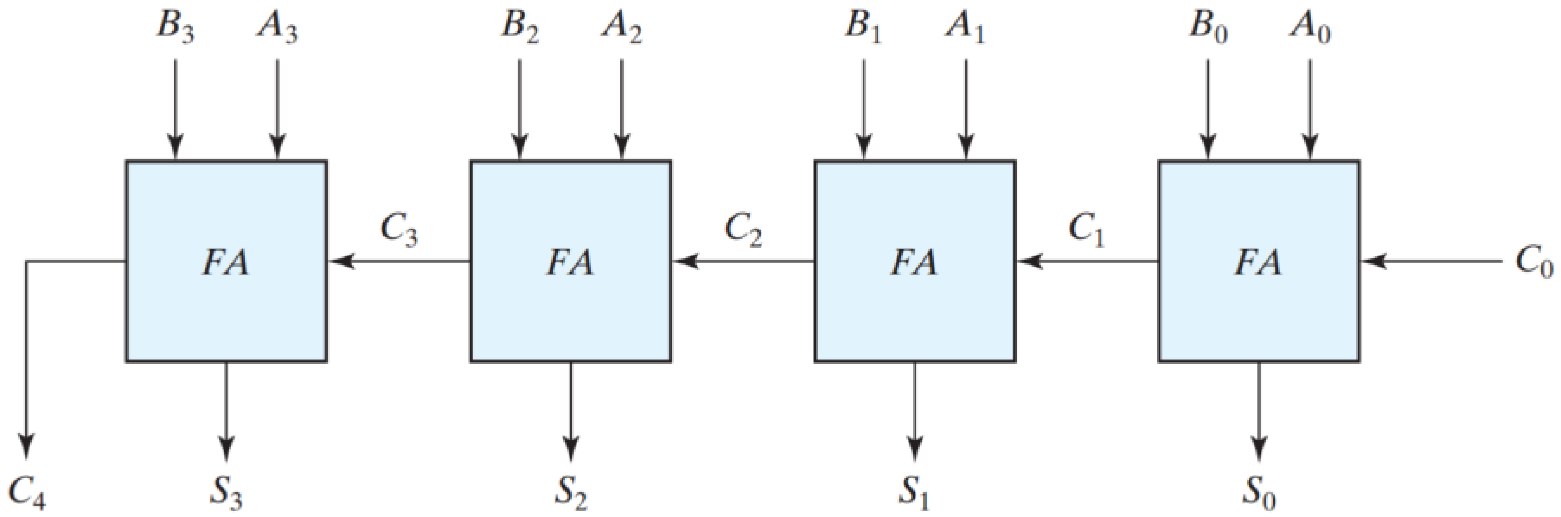
**Module Definition**

Module name: top_adder

**I/O Port Definitions**

| Port Name | Direction | Bus | MSB | LSB |
|---|---|---|---|---|
| InA | input | ☑ | 3 | 0 |
| InB | input | ☑ | 3 | 0 |
| OutSum | outp... | ☑ | 4 | 0 |

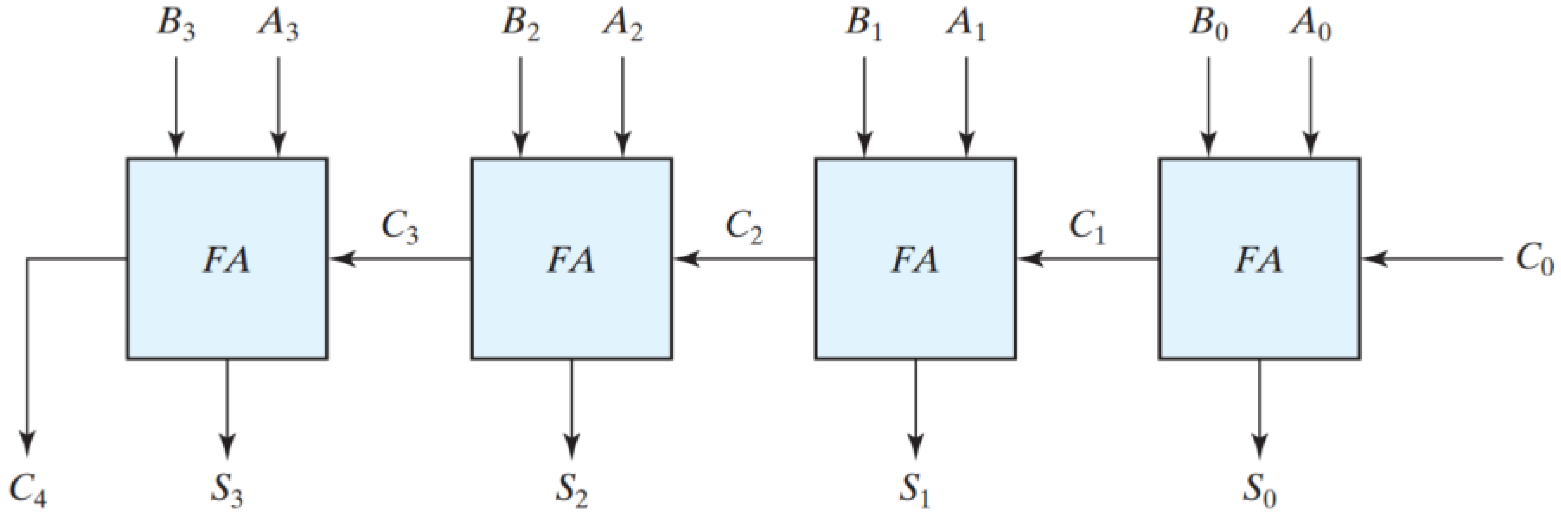
# 4-bit FA



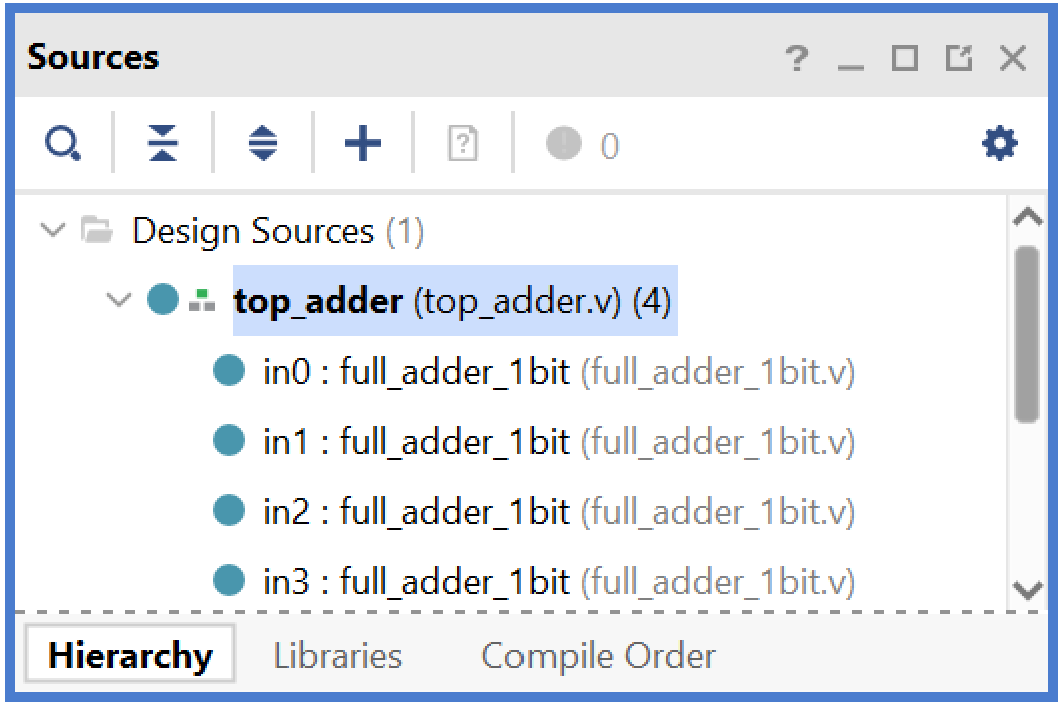
```
module top_adder(
    input [3:0] InA,
    input [3:0] InB,
    output [4:0] OutSum
    );
    
    wire carry1 , carry2, carry3;
    
    full_adder_1bit in0(.FA1_InA(InA[0]) , .FA1_InB(InB[0]) , .FA1_InC(1'b0) , .FA1_OutSum(OutSum[0]) , .FA1_OutC(carry1));
    full_adder_1bit in1(.FA1_InA(InA[1]) , .FA1_InB(InB[1]) , .FA1_InC(carry1) , .FA1_OutSum(OutSum[1]) , .FA1_OutC(carry2));
    full_adder_1bit in2(.FA1_InA(InA[2]) , .FA1_InB(InB[2]) , .FA1_InC(carry2) , .FA1_OutSum(OutSum[2]) , .FA1_OutC(carry3));
    full_adder_1bit in3(.FA1_InA(InA[3]) , .FA1_InB(InB[3]) , .FA1_InC(carry3) , .FA1_OutSum(OutSum[3]) , .FA1_OutC(OutSum[4]));

endmodule
```

# 4-bit FA



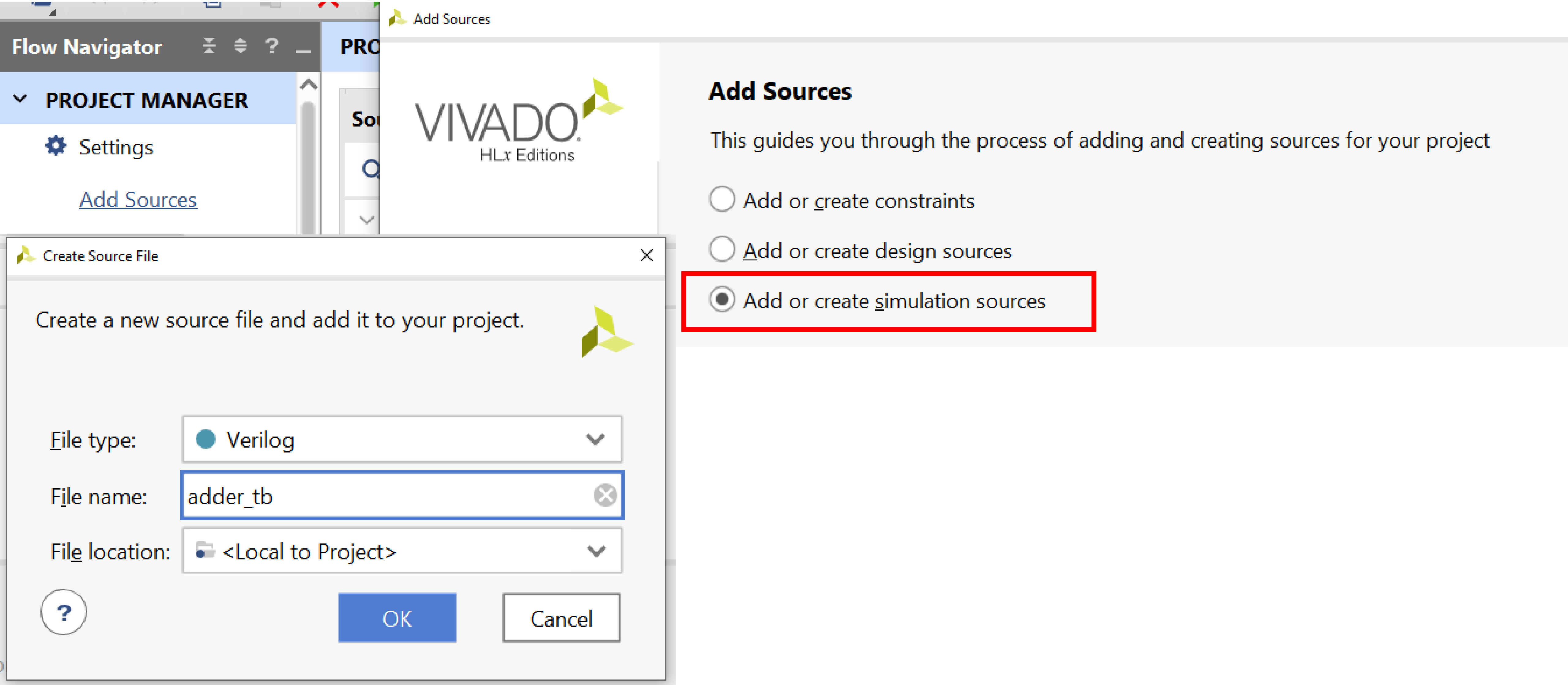
**PROJECT MANAGER** - Lab1_FA



Design Sources (1)

- **top_adder** (top_adder.v) (4)
  - in0 : full_adder_1bit (full_adder_1bit.v)
  - in1 : full_adder_1bit (full_adder_1bit.v)
  - in2 : full_adder_1bit (full_adder_1bit.v)
  - in3 : full_adder_1bit (full_adder_1bit.v)

# Testbench



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

Create Source File

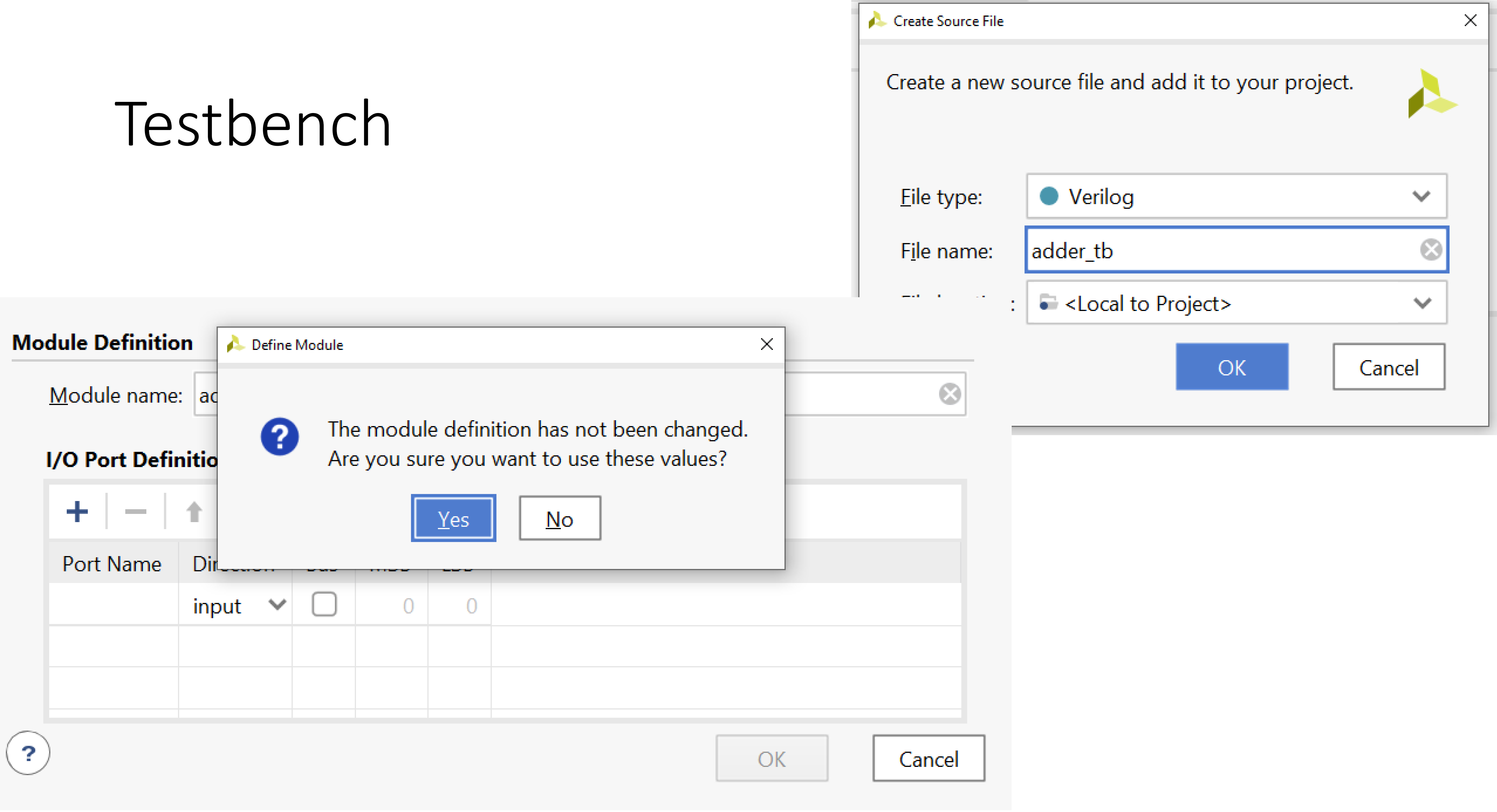
Create a new source file and add it to your project.

File type: Verilog

File name: adder_tb

File location: <Local to Project>

# Testbench



Create Source File

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

File type: ● Verilog

File name: adder_tb

: <Local to Project>

Module Definition

Module name: ac

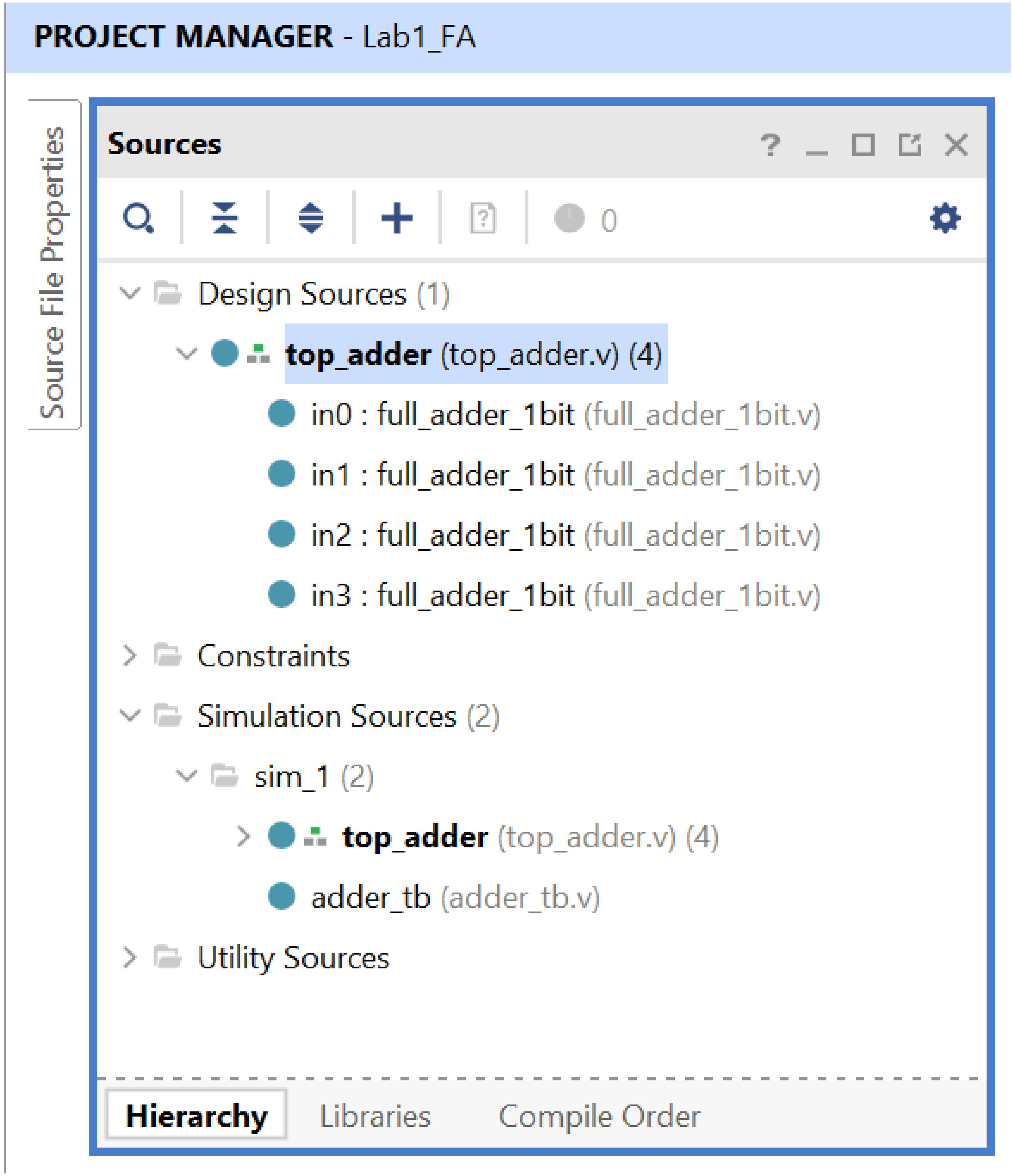
I/O Port Definitio

| Port Name | Direction | Bus | MSB | LSB |
|---|---|---|---|---|
| | input | □ | 0 | 0 |

Define Module

The module definition has not been changed.
Are you sure you want to use these values?

# Testbench



PROJECT MANAGER - Lab1_FA

Sources

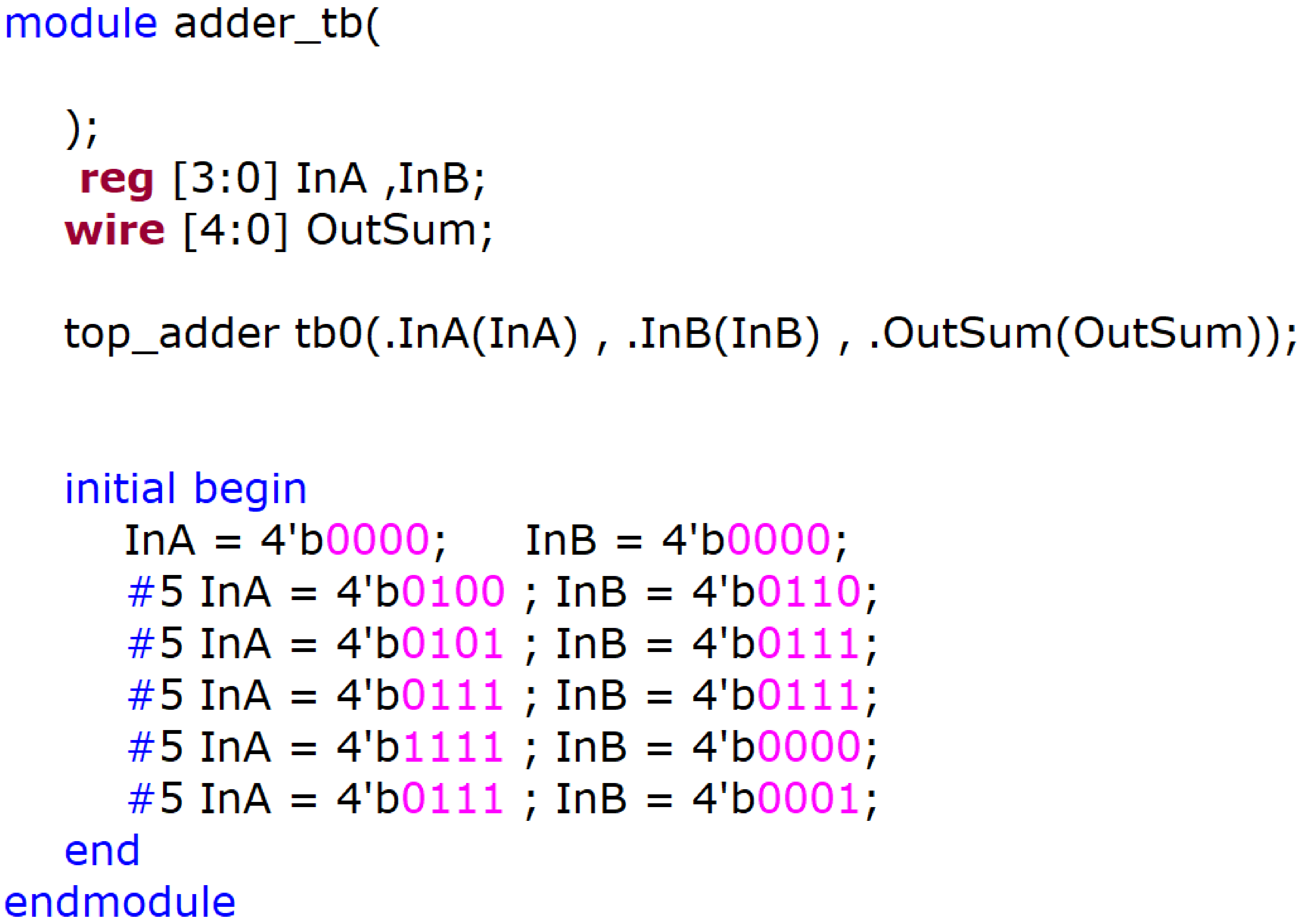
Design Sources (1)
- top_adder (top_adder.v) (4)
  - in0 : full_adder_1bit (full_adder_1bit.v)
  - in1 : full_adder_1bit (full_adder_1bit.v)
  - in2 : full_adder_1bit (full_adder_1bit.v)
  - in3 : full_adder_1bit (full_adder_1bit.v)

Constraints

Simulation Sources (2)
- sim_1 (2)
  - top_adder (top_adder.v) (4)
  - adder_tb (adder_tb.v)

Utility Sources

# Testbench



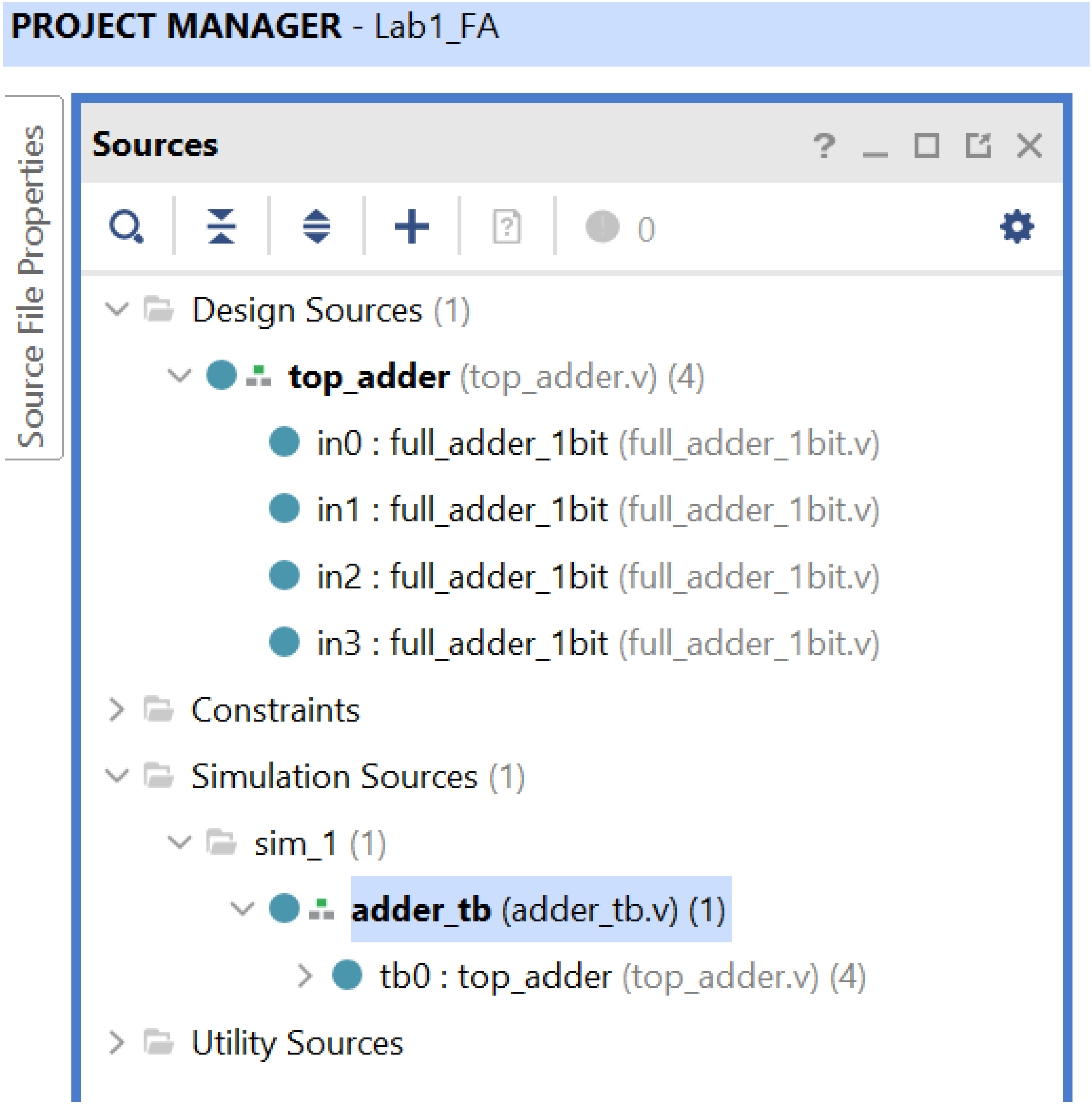
```
module adder_tb(

   );
    reg [3:0] InA ,InB;
   wire [4:0] OutSum;

   top_adder tb0(.InA(InA) , .InB(InB) , .OutSum(OutSum));


   initial begin
      InA = 4'b0000;     InB = 4'b0000;
      #5 InA = 4'b0100 ; InB = 4'b0110;
      #5 InA = 4'b0101 ; InB = 4'b0111;
      #5 InA = 4'b0111 ; InB = 4'b0111;
      #5 InA = 4'b1111 ; InB = 4'b0000;
      #5 InA = 4'b0111 ; InB = 4'b0001;
   end
endmodule
```

# Testbench



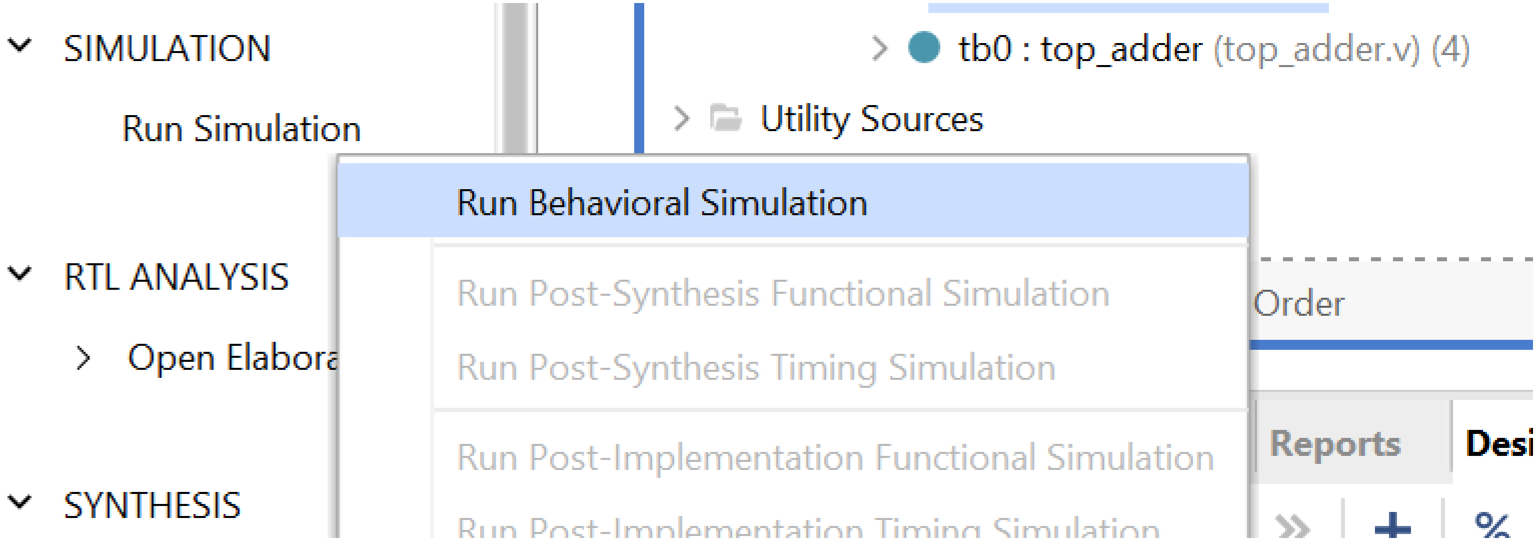
PROJECT MANAGER - Lab1_FA

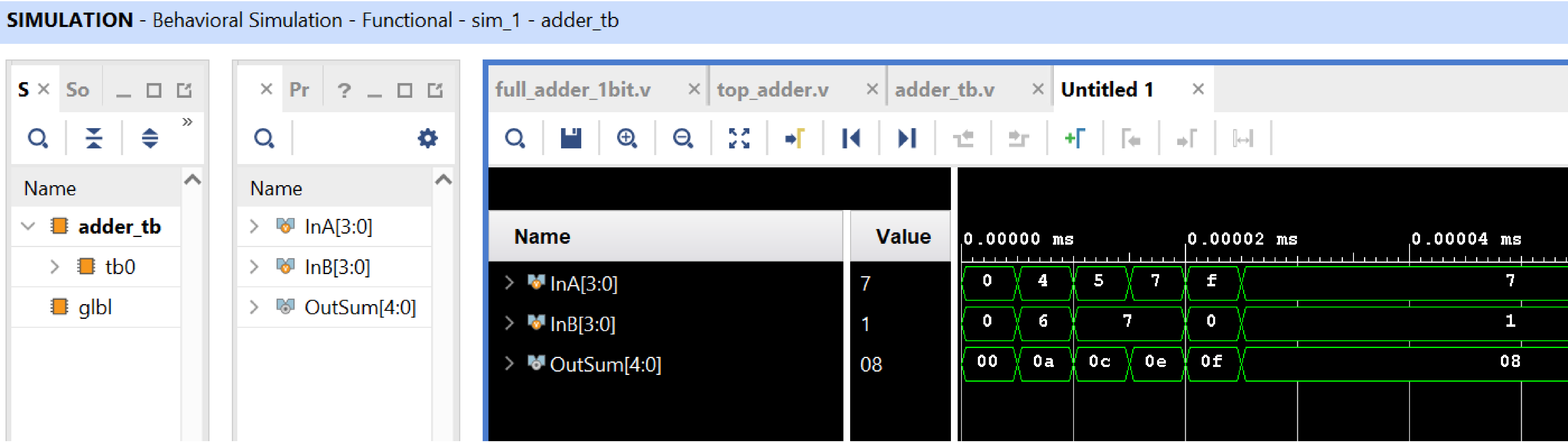
Source File Properties

Sources

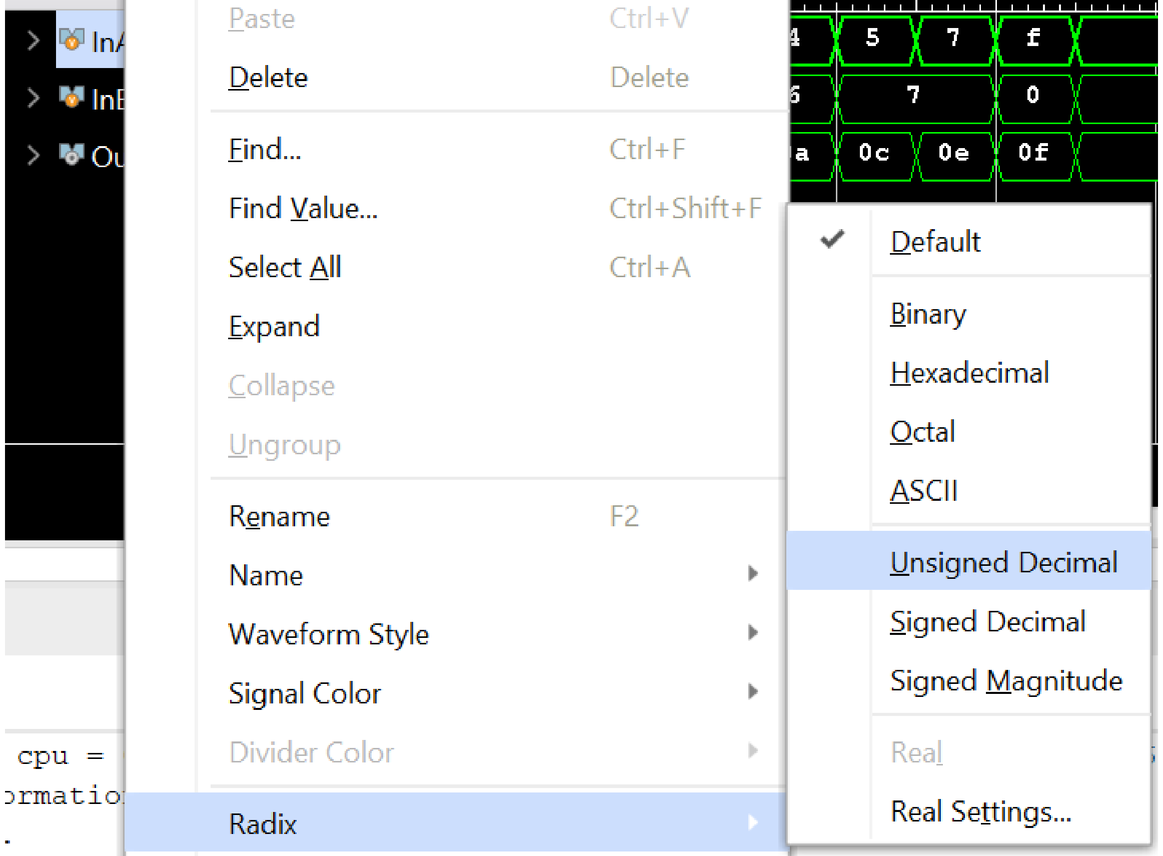
- Design Sources (1)
  - top_adder (top_adder.v) (4)
    - in0 : full_adder_1bit (full_adder_1bit.v)
    - in1 : full_adder_1bit (full_adder_1bit.v)
    - in2 : full_adder_1bit (full_adder_1bit.v)
    - in3 : full_adder_1bit (full_adder_1bit.v)
- Constraints
- Simulation Sources (1)
  - sim_1 (1)
    - adder_tb (adder_tb.v) (1)
      - tb0 : top_adder (top_adder.v) (4)
- Utility Sources

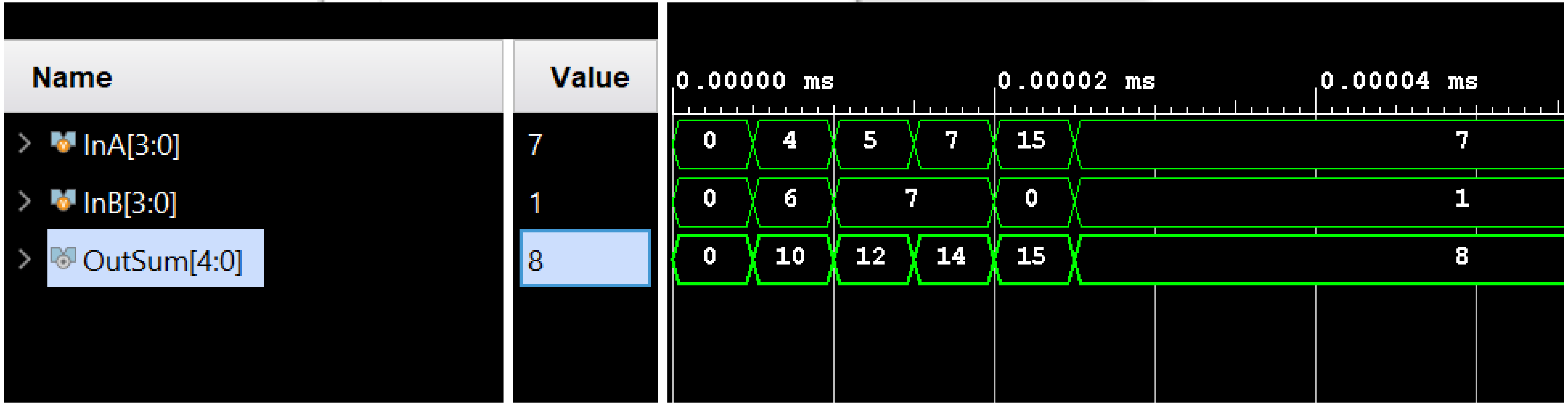
# Testbench


SIMULATION
Run Simulation
RTL ANALYSIS
Open Elabora
SYNTHESIS
tb0 : top_adder (top_adder.v) (4)
Utility Sources
Run Behavioral Simulation
Run Post-Synthesis Functional Simulation
Run Post-Synthesis Timing Simulation
Run Post-Implementation Functional Simulation
Run Post-Implementation Timing Simulation
Order
Reports
Desi


SIMULATION - Behavioral Simulation - Functional - sim_1 - adder_tb
S
So
Name
adder_tb
tb0
glbl
Pr
Name
InA[3:0]
InB[3:0]
OutSum[4:0]
full_adder_1bit.v
top_adder.v
adder_tb.v
Untitled 1
Name
Value
InA[3:0] 7
InB[3:0] 1
OutSum[4:0] 08
0.00000 ms
0.00002 ms
0.00004 ms
0 4 5 7 f 7
0 6 7 0 1
00 0a 0c 0e 0f 08

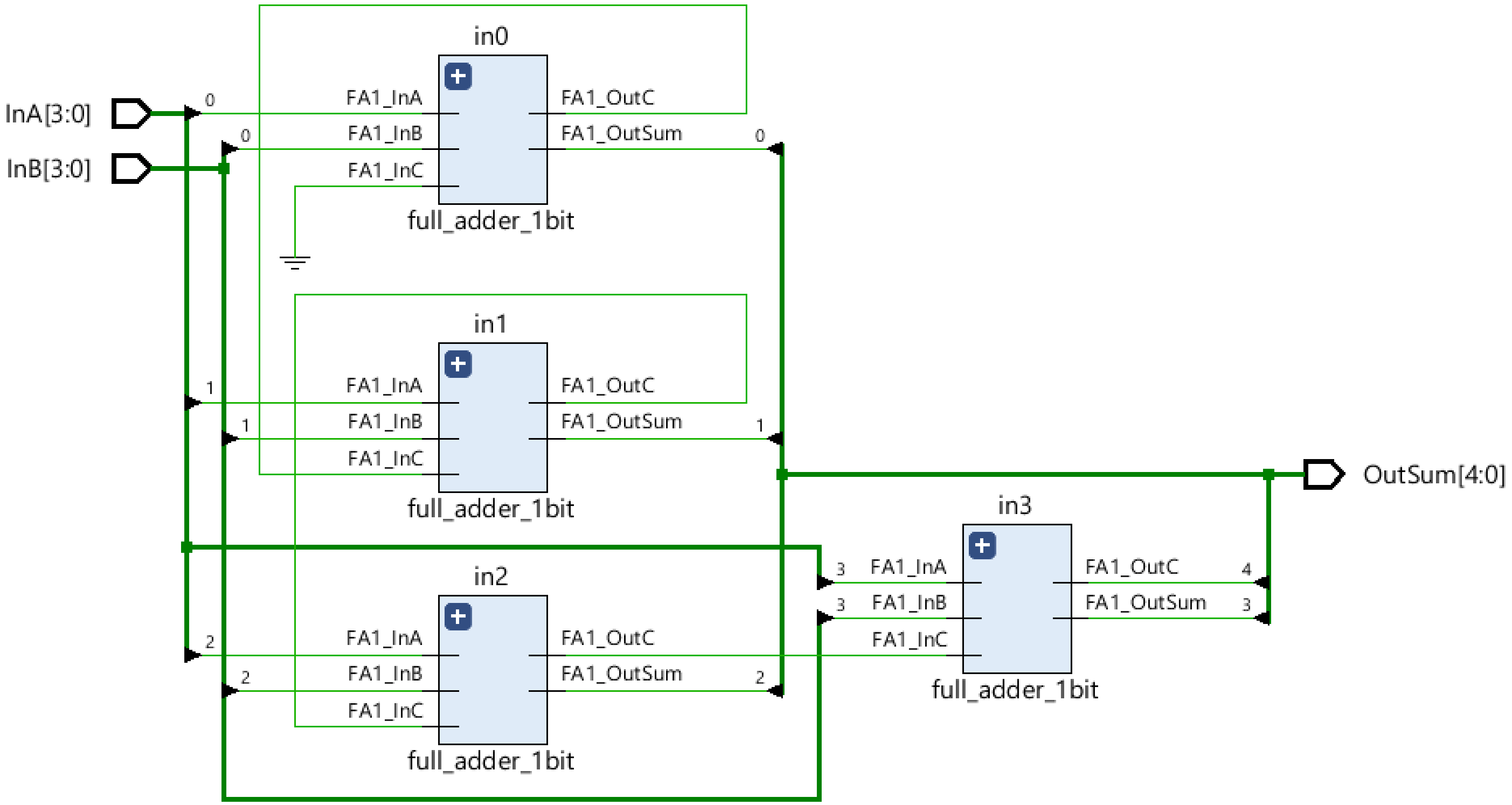
# Testbench



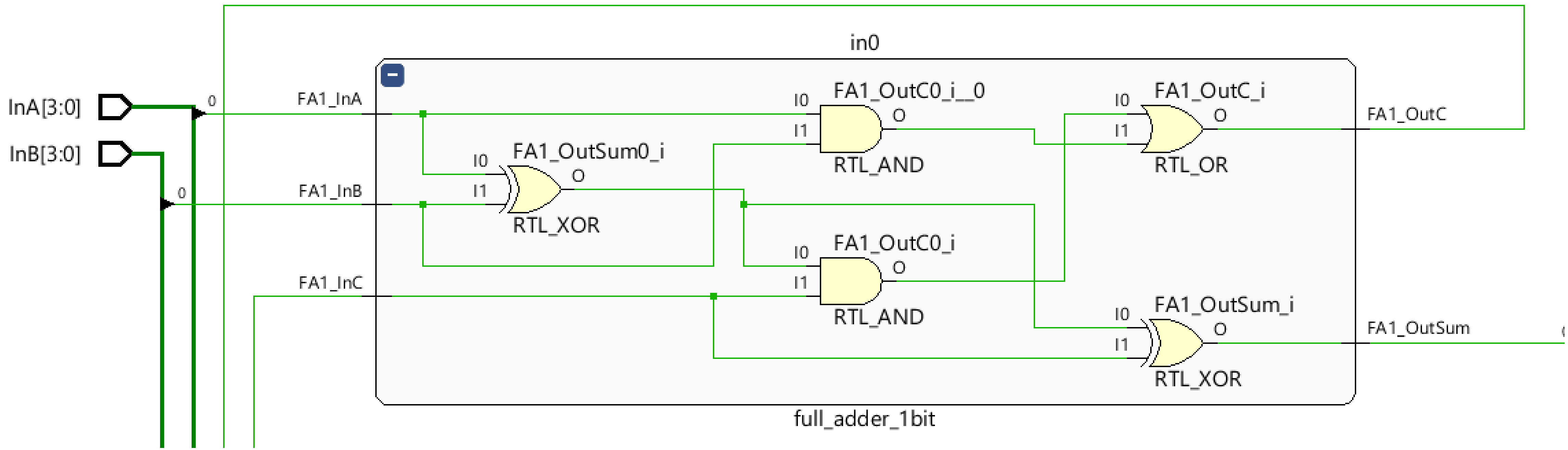


| Name | Value |
|---|---|
| InA[3:0] | 7 |
| InB[3:0] | 1 |
| OutSum[4:0] | 8 |

# Elaborated Design

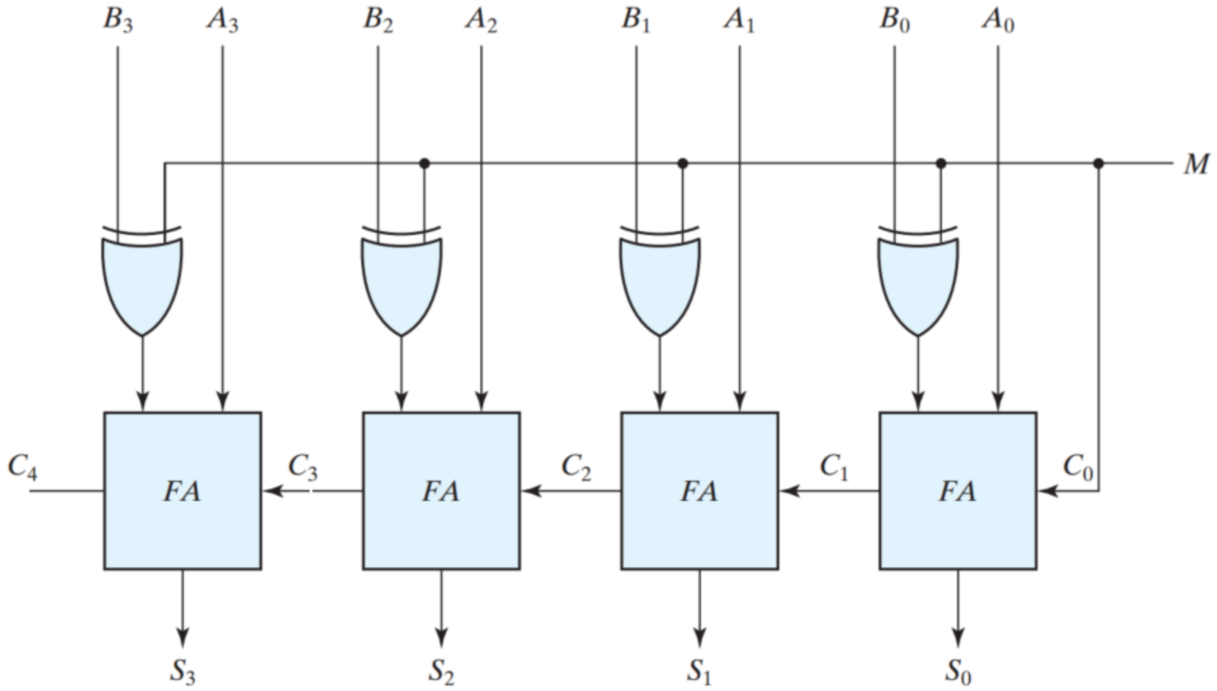


# Elaborated Design



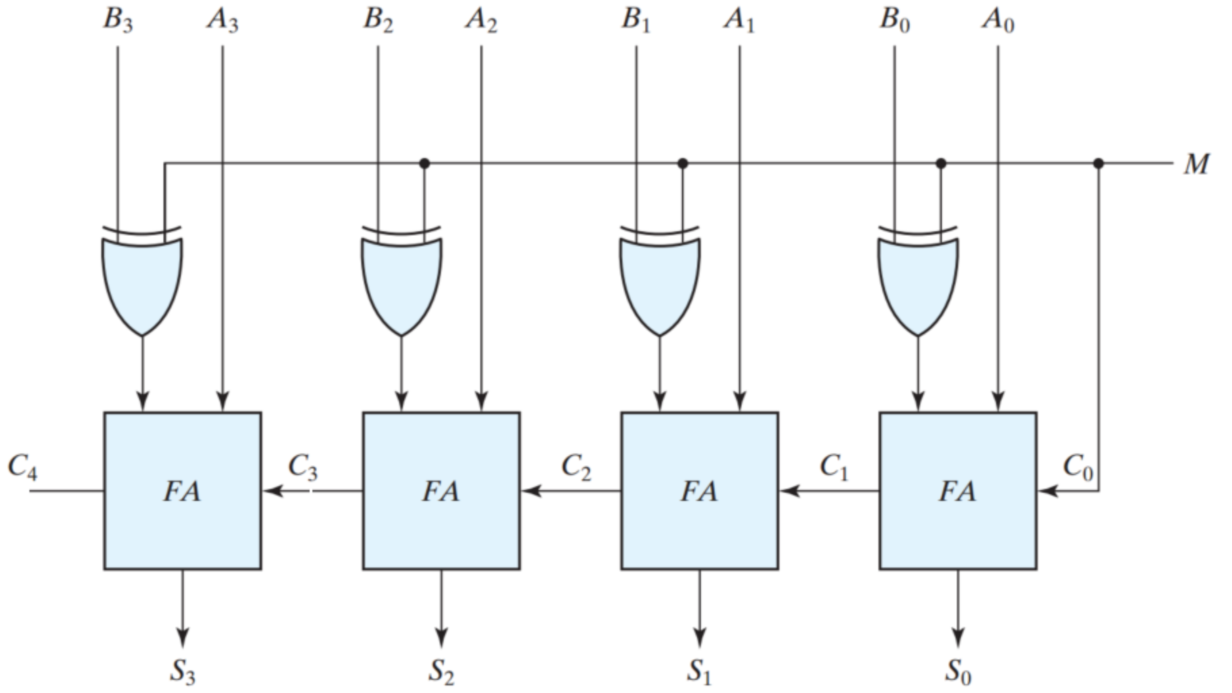
# Homework

# Adders



- For adder/subtractor with signed number inputs, add three independent output flags 1) First output flag goes high when overflow occurs, 2) Second output flag goes high when sum is negative, and 3) Third output flag goes high when sum is zero

# Overflow



$$\begin{array}{rr} (+7) & 0111 \\ + \ (-2) & +\ 1110 \\ \hline (+5) & 10101 \end{array}$$

$c_4 = 1$
$c_3 = 1$

$$\begin{array}{rr} (-7) & 1001 \\ +\ (+2) & +\ 0010 \\ \hline (-5) & 1011 \end{array}$$

$c_4 = 0$
$c_3 = 0$

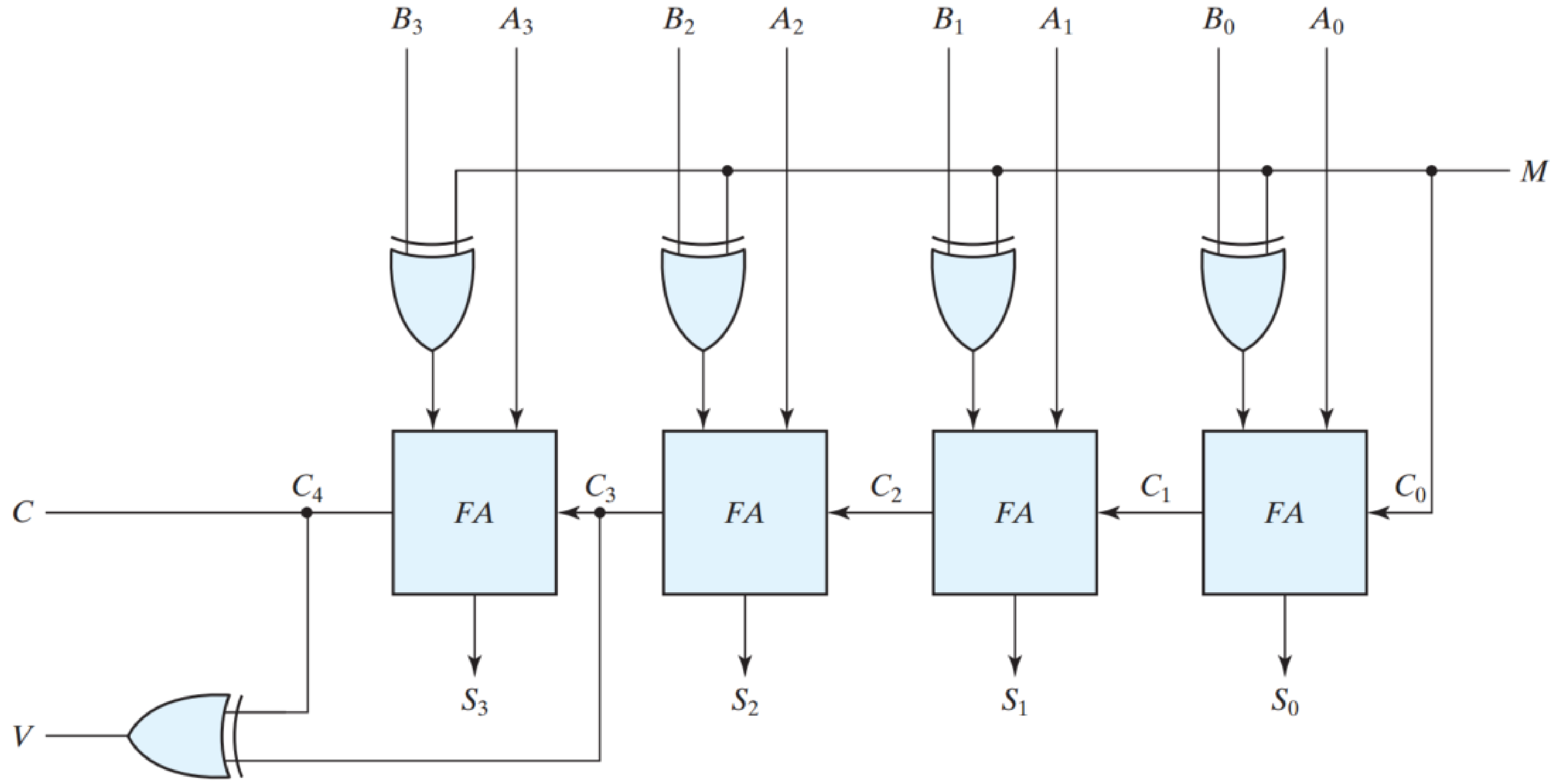
$$\begin{array}{rr} (-7) & 1001 \\ + \ (-2) & +\ 1110 \\ \hline (-9) & 10111 \end{array}$$

$c_4 = 1$
$c_3 = 0$

$$\begin{array}{rr} (+7) & 0111 \\ +\ (+2) & +\ 0010 \\ \hline (+9) & 1001 \end{array}$$

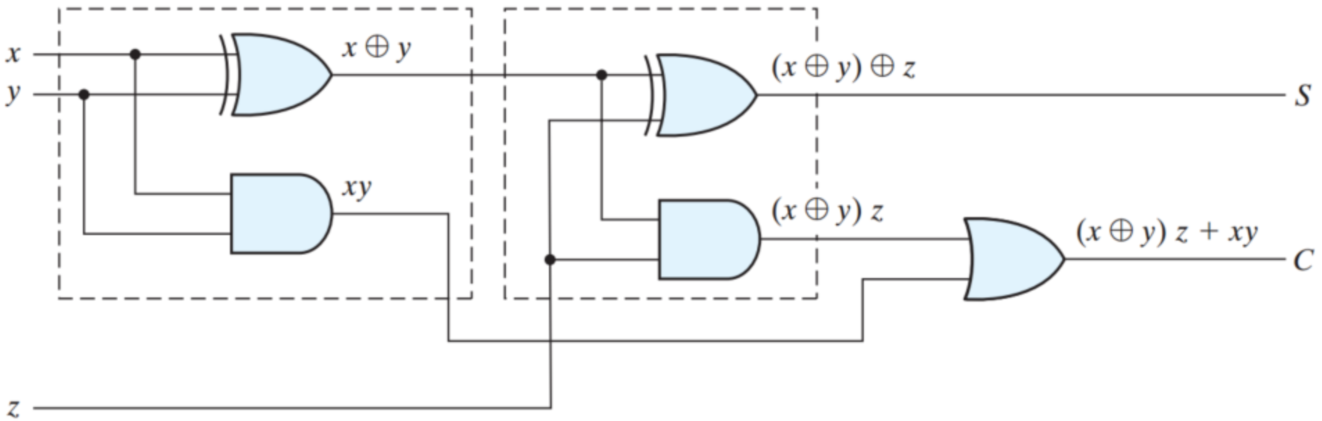
$c_4 = 0$
$c_3 = 1$

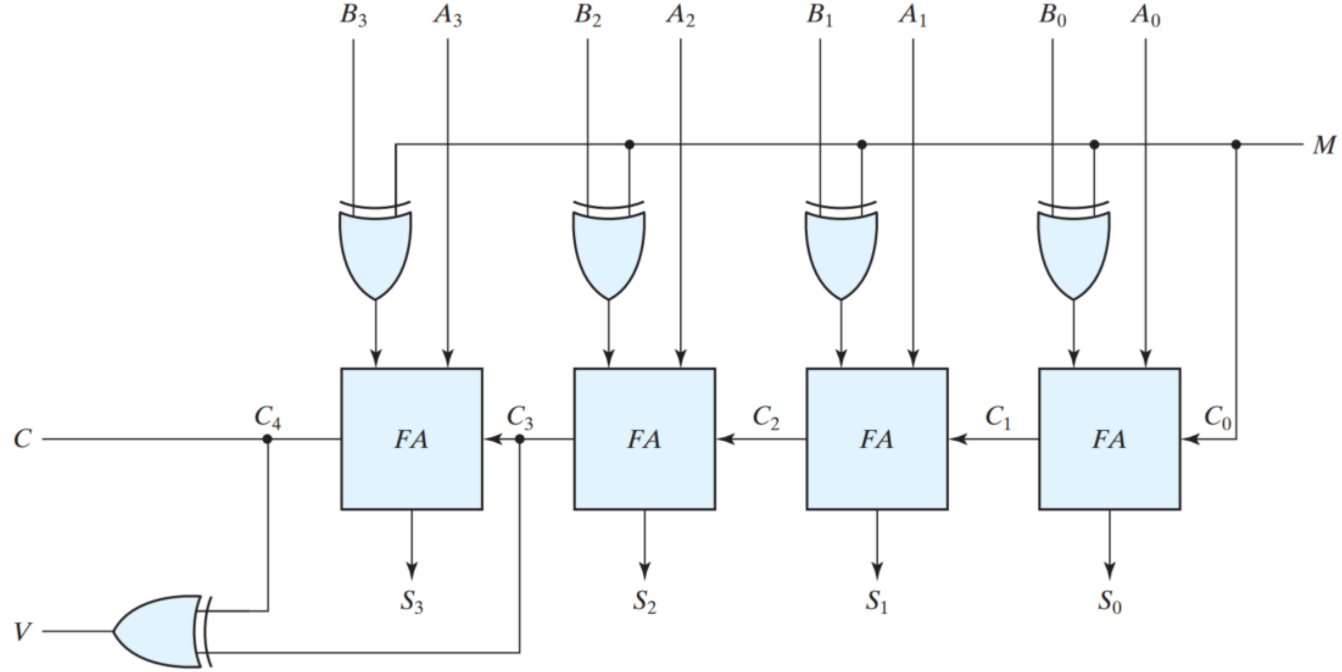
# Overflow



# Performance

- The longest propagation delay time in an adder is the time it takes the carry to propagate through the full adders.
- Inputs *A*3 and*B*3 are available as soon as input signals are applied to the adder. However, input carry *C*3 does not settle to its final value until *C*2 is available from the previous stage. Similarly, *C*2 has to wait for *C*1 and so on down to *C*0.

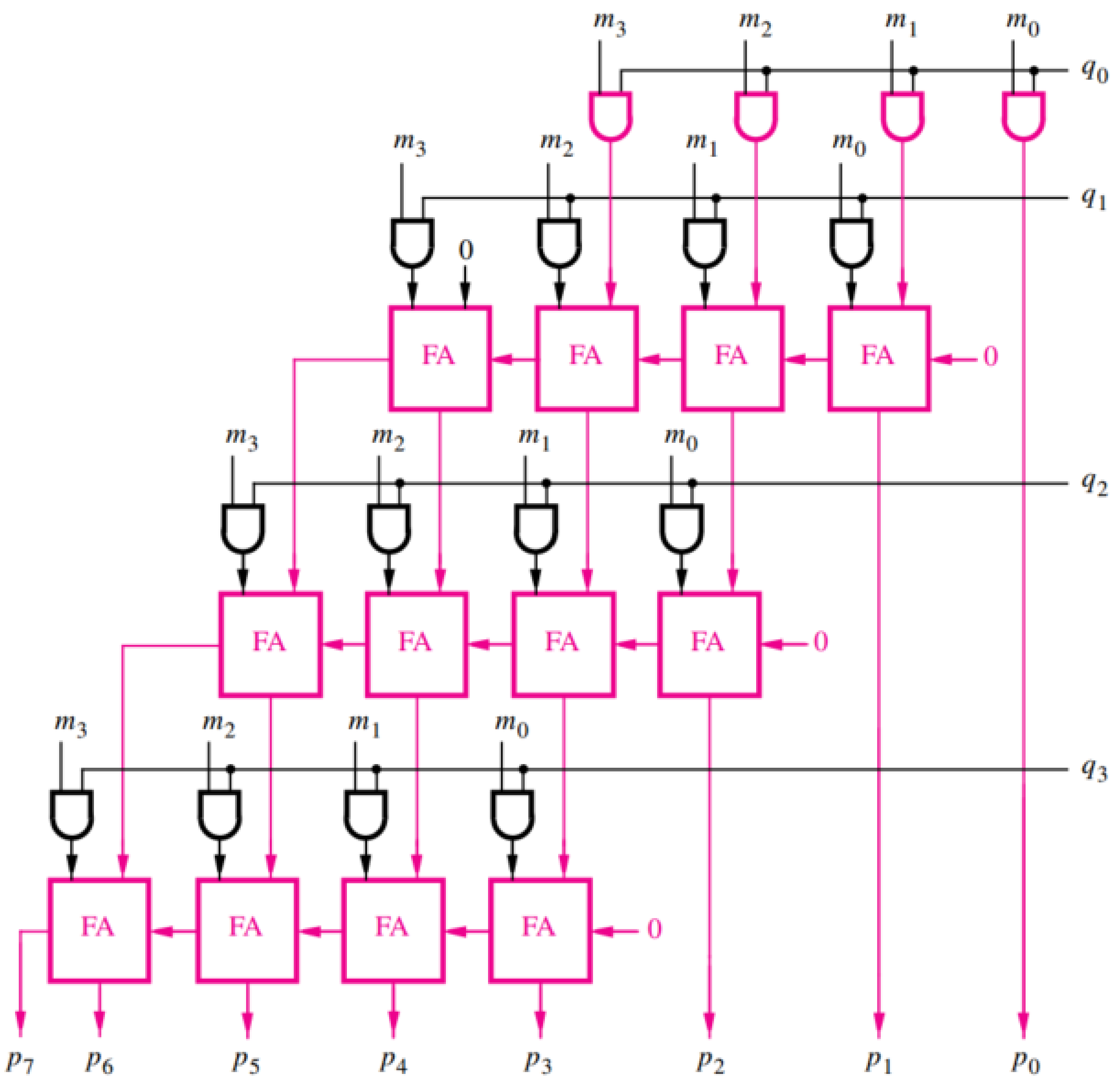




# Self Study

# Multiplication

$$\begin{array}{lcccccccc}
 & & & & & m_3 & m_2 & m_1 & m_0 \\
 & & & & \times & q_3 & q_2 & q_1 & q_0 \\ \hline
\text{Partial product 0} & & & & & m_3q_0 & m_2q_0 & m_1q_0 & m_0q_0 \\
 & & & & + \; m_3q_1 & m_2q_1 & m_1q_1 & m_0q_1 & \downarrow \\ \hline
\text{Partial product 1} & & & PP1_5 & PP1_4 & PP1_3 & PP1_2 & PP1_1 & \downarrow \\
 & & & + \; m_3q_2 & m_2q_2 & m_1q_2 & m_0q_2 & \downarrow & \downarrow \\ \hline
\text{Partial product 2} & & PP2_6 & PP2_5 & PP2_4 & PP2_3 & PP2_2 & \downarrow & \downarrow \\
 & & + \; m_3q_3 & m_2q_3 & m_1q_3 & m_0q_3 & \downarrow & \downarrow & \downarrow \\ \hline
\text{Product P} & p_7 & p_6 & p_5 & p_4 & p_3 & p_2 & p_1 & p_0
\end{array}$$


$m_3$
$m_2$
$m_1$
$m_0$
$q_0$
$q_1$
$q_2$
$q_3$
0
FA
$p_7$
$p_6$
$p_5$
$p_4$
$p_3$
$p_2$
$p_1$
$p_0$